Technical Report 468

C. J. Weinstein

Quantization Effects in Digital Filters

21 November 1969

Prepared under Electronic Systems Division Contract AF 19(628)-5167 by

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Lexington, Massachusetts

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LINCOLN LABORATORY

QUANTIZATION EFFECTS IN DIGITAL FILTERS

C. J. WEINSTEIN

Group 62

TECHNICAL REPORT 468

21 NOVEMBER 1969

LEXINGTON MASSACHUSETTS

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology, with the support of the Department of the Air Force under Contract AF 19(628)-5167.

QUANTIZATION EFFECTS IN DIGITAL FILTERS*

ABSTRACT

When a digital filter is implemented on a computer or with special-purpose hardware, errors and constraints due to finite word length are unavoidable. These quantization effects must be considered, both in deciding what register length is needed for a given filter implementation and in choosing between several possible implementations of the same filter design, which will be affected differently by quantization.

Quantization effects in digital filters can be divided into four main categories: quantization of system coefficients, errors due to analog-digital (A-D) conversion, errors due to roundoffs in the arithmetic, and a constraint on signal level due to the requirement that overflow be prevented in the computation.

The effects of these errors and constraints will vary, depending on the type of arithmetic used. Fixed point, floating point, and block floating point are three alternate types of arithmetic often employed in digital filtering.

A very large portion of the computation performed in digital filtering is composed of two basic algorithms – the first- or second-order, linear, constant coefficient, recursive difference equation; and computation of the discrete Fourier transform (DFT) by means of the fast Fourier transform (FFT). These algorithms serve as building blocks from which the most complicated digital filtering systems can be constructed.

The effects of quantization on implementations of these basic algorithms are studied in some detail. Sensitivity formulas are presented for the effects of coefficient quantization on the poles of simple recursive filters. The mean-squared error in a computed DFT, due to coefficient quantization in the FFT, is estimated. For both recursions and the FFT, the differing effects of fixed and floating point coefficients are investigated. Statistical models for roundoff errors and A-D conversion errors, and linear system noise theory, are employed to estimate output noise variance in simple recursive filters and in the FFT. By considering the overflow constraint in conjunction with these noise analyses, output noise-to-signal ratios are derived. Noise-to-signal ratio analyses are carried out for fixed, floating, and block floating point arithmetic, and the results are compared.

* This report is based on a thesis of the same title submitted to the Department of Electrical Engineering at the Massachusetts Institute of Technology on 31 July 1969 in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

All the noise analyses are based on simple statistical models for roundoff errors (and A-D conversion errors). Of course, somewhat different models are applied for the different types of arithmetic. These models cannot in general be verified theoretically, and thus one must resort to experimental noise measurements to support the predictions obtained via the models. A good deal of experimental data on noise measurements is presented here, and the empirical results are generally in good agreement with the predictions based on the statistical models.

The ideas developed in the study of simple recursive filters and the FFT are applied to analyze quantization effects in two more complicated types of digital filters – frequency sampling and FFT filters. The frequency sampling filter is realized by means of a comb filter and a bank of second-order recursive filters; while an FFT filter implements a convolution via an FFT, a multiplication in the frequency domain, and an inverse FFT. Any finite duration impulse response filter can be realized by either of these methods. The effects of coefficient quantization, roundoff noise, and the overflow constraint are investigated for these two filter types. Through use of a specific example, realizations of the same filter design, by means of the frequency sampling and FFT methods, are compared on the basis of differing quantization effects.

Accepted for the Air Force
Franklin C. Hudson
Chief, Lincoln Laboratory Office

CONTENTS

ACKNOWLEDGMENTS

Dr. Bernard Gold provided outstanding guidance and assistance throughout the course of this research effort. He supplied many specific technical suggestions, and perhaps more importantly gave, at various times of uncertainty, perceptive advice concerning the direction the research should take.

My discussions with Professor Yuk-Wing Lee helped supply perspective as to how my research fits into the broader context of signal and system theory. Professor Alan V. Oppenheim worked with me actively on parts of the research, as evidenced by our two joint publications. Throughout the work, he supplied many useful suggestions. Mr. Charles M. Rader provided many perceptive comments and ideas; our frequent discussions contributed both to the report effort and to my knowledge of digital signal processing.

All the experimental work reported was carried out on the M. I. T. Lincoln Laboratory Univac 1219 computer facility. Throughout the period of the research, I have been supported by a National Science Foundation fellowship.

QUANTIZATION EFFECTS IN DIGITAL FILTERS

I. INTRODUCTION

In recent years, there has been a great upsurge of interest in digital filtering, or the process of spectrum shaping or spectrum analysis via digital operations on the samples of a signal. Some important factors motivating this concern have been: (a) the widespread use of computer simulation in the design of analog systems; (b) application of the general-purpose computer to complex signal processing tasks, such as processing of seismic data or of pictures; and (c) increasing speed and decreasing cost and size of digital circuitry, making real-time digital filters feasible and practical. In addition, the discovery of the fast Fourier transform and its application to high speed convolution has added a new dimension to the field of digital filtering, allowing efficient realization of filter designs previously considered impractical due to the excessive computation time needed for their implementation.

The task of obtaining a digital filter for a specified purpose can be divided roughly into two phases – design and implementation. In design, as in the approximation problem for the analog case, one seeks a rational transfer function of finite order which approximates to a satisfactory tolerance some desired frequency response characteristic (or one may seek to approximate some desired time response). In implementation, or synthesis, one seeks a numerical algorithm or hardware configuration which will efficiently implement this transfer function. Clearly, the design and implementation phases are not quite independent, as the designer, given a choice, would seek those designs which could be most efficiently implemented.

This report will focus on an issue of central concern in digital filter implementation, namely, the problem of quantization in digital filters. Whether a digital filter is implemented on a general-purpose computer or with special-purpose hardware, finite register length arithmetic must be used. One must take careful account of the quantization inherent in this arithmetic, both in deciding what register length is needed for a given implementation and in choosing between several possible implementations of the same filter design, which will be affected differently by quantization.

To illustrate a few of the options available for implementing a given filter design, consider a digital transfer function

$$H(z) = \frac{\sum_{i=1}^{M} a_i z^{-i}}{1 + \sum_{i=1}^{N} b_i z^{-i}} \tag{1-1}$$

where the designer has specified the a_i, b_i, N (the number of poles), and M (the number of zeros, excluding the trivial zeros at $z = 0$ and $z = \infty$). This filter can be realized via a single N^{th}-order recursive difference equation

$$y(n) = \sum_{i=1}^{N} b_i y(nT - iT) + \sum_{i=0}^{M} a_i x(nT - iT) \qquad \text{(I-2)}$$

where the current output $y(nT)$ is obtained from a linear combination of past outputs, and current and past inputs (T is the sampling interval). This realization is usually called the direct form. As an alternative, one can factor the rational fraction in (I-1) in terms of its zeros q_i and poles p_i as

$$H(z) = K \frac{\prod_{i=1}^{M} (1 - q_i z^{-1})}{\prod_{i=1}^{N} (1 - p_i z^{-1})} \qquad \text{(I-3)}$$

and synthesize the filter as a cascade of simple (first- or second-order) recursions, each corresponding to a factor or group of factors in (I-3). Another possible form of realization could be obtained by expanding $H(z)$ as a partial fraction sum, and synthesizing the filter as a parallel combination of simple filters representing the terms in the sum.

There are many other types of algorithms which can be used to implement a given $H(z)$. In addition, there are various types of arithmetic (e.g., fixed or floating point) which can be used. All realizations would yield the same filter, except for the fact that the errors due to quantization are different for the various realizations.

A. OUTLINE OF PROBLEM AREA

Broadly, the aim of this report is to study the effects of quantization in the various types of digital filter implementations, and thus to provide insight into how one efficiently implements digital filters in view of quantization problems. To set a framework for the study, we need to define more explicitly this rather broad problem area. To this end, we now outline (1) the basic computational algorithms which are used in digital filtering, (2) the various types of arithmetic which are used in the computation, and (3) the various quantization effects which occur when these algorithms are implemented on a finite word length machine.

A basic numerical algorithm for digital filtering is the first- or second-order linear, constant coefficient, recursive difference equation. Usually, one would be wise to construct a complicated recursive digital filter (having poles, as well as zeros, in its transfer function) as a combination of first- and second-order equations, since such a realization has been found to be less sensitive to quantization effects than a realization using, say, a single higher-order difference equation. The second basic numerical algorithm for digital filtering is the fast Fourier transform (FFT), which permits rapid computation of the discrete Fourier transform (DFT) of a signal. This algorithm can be used, for example, to perform digital spectrum analysis, or to implement any nonrecursive digital filter via the route of fast convolution. A very large portion of the computation performed in digital filtering, and, in fact, in digital signal processing, is composed of these two basic algorithms – the simple recursive difference equation and the FFT. Thus, a sizable portion of this report will be devoted to studying the effects of quantization on realization of these algorithms.

Two basic types of arithmetic – fixed and floating point – can be used in the implementation of these digital filtering algorithms. The errors and constraints due to finite word length are different for these two types of arithmetic, and both will be studied here. In addition, a hybrid scheme known as block floating point, where a single exponent is used for an entire block of numbers, has been found useful for both the FFT and recursive difference equations. The quantization effects for block floating point arithmetic also will be studied.

The effects of quantization on digital filter performance can be divided into four key categories. The first three can be classified as quantization errors, and the fourth as a constraint imposed by the finite word length. All the quantization effects to be listed occur both in difference equations and in FFT implementation, and affect filter performance (in different ways) for any of the three types of arithmetic mentioned above. The first class of quantization errors we will mention is the error caused by discretization of the fixed system coefficients – the constant multipliers in our difference equations or the sine-cosine table for the FFT. Usually, a filter is designed on the basis of perfectly accurate coefficients; the quantization of the coefficients to finite word length means that, in practice, we will construct a slightly different filter, just as when a precisely designed analog filter is built from imprecise components. If our digital filter is used to process an analog signal, a second type of quantization error is incurred in the analog-digital (A-D) conversion of the signal. The third source of quantization errors is the roundoff of the results of multiplications and additions used in the filter implementation. A-D conversion and roundoff errors are quite similar in that both can, for many practical cases, be analyzed as additive noise superposed on an otherwise linear system. The fourth important quantization effect is the limit imposed by the finite word length on the range of numbers which can be represented in the filter. This limit forces us to arrange our filter implementation to prevent overflow. This constraint is much more severe in fixed point than in floating or block floating point, where the presence of an exponent allows for a much larger dynamic range. The limit on signal level due to the overflow constraint must be considered, together with the noise of roundoff and A-D conversion, to determine the noise-to-signal ratio we would expect at the output of a digital filter.

This completes our brief summary of the basic problem area of quantization in digital filters. A large portion of this report is devoted to a study of the various quantization effects in simple recursive filters and FFT's. In the remainder of the report, the ideas developed are applied to analyze quantization effects in two more complicated types of digital filters – frequency sampling filters and FFT filters. From an input-output viewpoint, these are both nonrecursive type filters, which is equivalent to saying that they have only zeros in their transfer function or have impulse responses which are nonzero only for a finite period of time. But the actual realization of the two filter types is quite different, as the frequency sampling filter utilizes a comb-filter and a band of second-order recursive filters, while an FFT filter implements a convolution via an FFT, a multiplication in the frequency domain, and an inverse FFT. One of the reasons for selecting these particular filter types for study is that any finite duration impulse response filter can be realized either via the frequency sampling method or via the FFT. These two alternate realizations of the same filter can be compared on the basis of differing quantization effects. In addition, excellent filter designs can be implemented via either technique, and quantization problems in both are fairly severe and quite interesting.

The problem area, as defined above, can be summarized in the following outline of key topics:

Basic Computational Algorithms

First- and second-order recursive difference equations
Fast Fourier transform

Types of Arithmetic

Fixed point
Block floating point
Floating point

Quantization Effects

Coefficient quantization
Analog-to-digital converter noise
Roundoff errors in computation
Overflow constraint

Filter Types to be Studied

Frequency sampling filters
FFT filters (high-speed convolution)

B. REVIEW OF PREVIOUS WORK

In this section, we report on previous work relevant to a study of quantization in digital filters. In addition to reporting on specific studies of the quantization problem, some background material (digital filter design, description of the FFT algorithm, quantization theory) is referenced. An attempt is made to indicate where the work to be described in this report conforms with previous studies.

1. Background on Digital Filters and FFT

Kaiser[1] has reviewed in some detail the history of digital filters, and presents a large and thorough bibliography. He discusses in detail various filter design techniques and quantization effects, with particular attention to coefficient quantization. A book by Gold and Rader[2] provides a thorough introduction to the whole field of digital signal processing. Their chapter on quantization effects is an excellent overview of these problems. Also included in the book is a development of the basic theory needed to analyze digital linear systems, techniques for digital filter design including a description of frequency sampling filters, a chapter explaining the FFT algorithm, and a chapter (written by Stockham) describing how the FFT may be used to implement convolution. Two basic and thorough papers on digital filter design techniques have been written by Golden and Kaiser[3] and Rader and Gold.[4] The original disclosure of the FFT algorithm was by Cooley and Tukey,[5] and a later paper by Cochran, et al.,[6] describes and elucidates many of the various forms of the algorithm. Stockham's paper[7] is the earliest presentation of the techniques of digital filtering via the FFT.

2. Coefficient Quantization

Previous work on the effects of coefficient quantization has concentrated on the coefficient sensitivity problem for filters realized via recursive difference equations. Most of the work done is applicable to either fixed or floating point filters. The relationship of changes in filter parameters (e. g., pole positions) to changes in the coefficients is the same for either fixed or floating point, although the actual errors due to quantizing the coefficients may be different.

Kaiser[8] studied the sensitivity to coefficient quantization of the pole positions of a digital filter realized via a single Nth-order difference equation. He concluded that, at least for low-pass or bandpass filters with tightly clustered poles, the sensitivity of the pole positions increased markedly with the order N of the equation. By an approximate analysis he indicates, for example, that the number of bits which must be retained in the filter coefficients to prevent instability can grow linearly with N. On the basis of these results, Kaiser suggests that to avoid severe changes from a desired frequency response function, or even instability, one should use a cascade or parallel arrangement of low-order recursions (rather than a single high-order recursion) to synthesize any reasonably complex filter with sharp transitions between pass and stop bands.

Rader and Gold[9] considered in detail the coefficient sensitivity problem for a second-order filter, and presented a realization via a pair of coupled first-order equations which was less sensitive to coefficient quantization than a single second-order equation. This coupled form can be used as a building block for a cascade or parallel realization of a more complex system.

Knowles and Olcayto[10] present a very interesting approach to the coefficient quantization problem, modeling the actual finite precision filter as a parallel combination of the original infinite precision transfer function, and a stray transfer function to account for coefficient quantization. They apply their model to analyze the effects of coefficient quantization for direct, cascade, and parallel form filter realizations. They carry out a series of experiments on a particular filter design (realized via direct, cascade, and parallel forms) to check the predicted deviation between the frequency responses of the finite and infinite precision filters. The agreement of experiment with theory is excellent. For the filter example they consider, the direct form is by far the most sensitive to coefficient quantization, and thus requires the most bits for a satisfactory realization. Thus, for this example, the work of Kaiser is corroborated.

Mantey[11] studies the coefficient quantization problem from the viewpoint of trying to select a state variable representation for the filter which is least sensitive to coefficient quantization. The state variable representations he compares are essentially the direct, cascade, and parallel forms, and his results also indicate that a cascade or parallel realization is less sensitive to coefficient quantization than a direct realization.

The key conclusion of all these workers seems to be that in order to minimize coefficient quantization problems, implementation of high-order difference equations should be avoided. Complicated filters should be implemented via combinations of simple difference equations.

An interesting application of some of the ideas of these previous workers is the study of coefficient quantization in frequency sampling filters. In addition to the present description of work on this problem, I have also reported upon it previously.[12]

With respect to the problem of coefficient quantization in the FFT, or in FFT filters, little previous work has been done. Some work on this problem will be reported in Secs. III-A and V-A.

3. Quantization Noise Models

Previous analyses of the effects of noise (due either to A-D conversion or roundoff) in digital filters have generally been based on a simple statistical model. That is, it has been assumed that errors of roundoff or A-D conversion have a uniform first-order probability density function, are uncorrelated from sample to sample, and are uncorrelated with the signal; and that all the noise sources in the digital network are independent of one another. These assumptions, though false for many pathological cases, have been supported by previous workers for a large class of signals.

Bennett,[13] by a somewhat involved analysis, derived the correlation function of quantization noise for Gaussian signals. He showed that for quantization steps small compared with signal variance, there is virtually no correlation between successive noise samples except when the correlation between successive signal samples is very high. He presents experimental results in which the spectrum of quantization noise was measured for quantization of sine waves and of random signals. In both cases, the assumption of white quantization noise was justified for sufficiently small quantization steps.

Widrow[14] presented a formalism by which the statistics of the noise due to quantization of random signals could be analyzed, given the statistics of the signals. The key to the approach was in viewing quantization as a sampling process on the probability density function of the input. He carried the analysis through for Gaussian signals, and corroborated the results of Bennett.

Knowles and Edwards[15] worked out theoretically the autocorrelation function of the noise introduced by quantizing a sine wave with random phase. The key step in their analysis is an expansion of the second-order probability density function of the input in a series of orthogonal functions. The results, which they verify experimentally, are that even for very coarse quantization (4 or 5 bits) the error spectra are essentially white. The signals flowing through a digital filter will quite often be of a periodic rather than, say, a Gaussian nature, and this result somewhat justifies the assumption of white quantization noise in such a case.

Sornmoonpin[16] presents a great deal of experimental evidence corroborating the simple white noise model for roundoff errors.

These previous studies relate to statistical models for fixed point roundoff errors, and not to floating point roundoff errors. The floating point case is more complicated, since the expected magnitude of roundoff errors depends on signal amplitude, and thus is not signal independent as in the fixed point case. Basic error models for floating point computation are presented in Wilkinson.[17] As will be seen, a statistical version of these models can be used successfully to predict roundoff noise in digital filters.

4. Statistical Theory of Noise Measurements

Although there is some solid basis for the simple noise models mentioned above, they are not valid, in general, and can be contradicted by counter examples. Ultimately, one must resort to experiment to verify the predictions of the models. The measurements are statistical in nature, and their understanding is based on a statistical theory of noise measurements.

A strong background in this theory, including the theory of spectral estimation for the discrete case, is presented in Grenander and Rosenblatt.[18] Some modern and practical procedures, by which one might use a digital computer for estimating noise correlation functions and spectra, are presented in Cooley, Lewis, and Welch.[19] These methods center on use of the FFT algorithm for noise analysis.

5. Noise in Digital Filters

Starting with the basic model discussed above, several workers have analyzed the effects of A-D conversion noise and roundoff noise on digital filter performance. Generally, noise measurements have been used to verify the theory. Noise effects have been studied both for recursive difference equation filters and, to a lesser extent, for the FFT. Although most previous work has concentrated on the fixed point case, some work on floating point and block floating point has been done.

The first three papers to be discussed deal with quantization noise in fixed point recursive filters.

Knowles and Edwards[20] used the white model for quantization, and linear system noise theory, to examine noise effects in direct, cascade, and parallel recursive filter implementations. They worked out bounds on the expected mean-square output noise in each type of realization, and carried out noise measurements which, to a reasonable extent, verified the theory. Care was taken to obtain statistically convergent noise measurements. No general statement was ventured about the realization forms, since only a few filter examples were considered in detail.

Gold and Rader[21] used the same model to derive convenient formulas for evaluating total (accounting for all noise sources) mean-squared output noise in digital filters. They analyze carefully the noise effects in first- and second-order filters, and present experimental results. Analysis of the computation time necessary for a statistically consistent measurement of output noise power is carried out. It is shown that, for very lightly damped filters, this measurement time becomes prohibitive.

Gold and Rabiner[22] carry out detailed analyses, both theoretical and experimental, of noise effects in a digital format synthesizer, realized via a cascade of digital resonators. An attempt is made to select that arrangement which produces acceptable vowel outputs, for the least register length. Overflow constraints and signal dynamic range requirements are considered, along with the noise problems, in selecting among various possible filter arrangements.

To complete the analysis of noise effects in a fixed point filter, it is necessary that one consider, along with the noise, the limit on signal level due to the overflow constraint. Later, in Sec. II-C, a formal approach to this problem will be presented and output noise-to-signal ratios for digital filters will be derived.

An analysis of roundoff errors in floating point recursive filters has been presented by Sandberg.[23] As mentioned above, floating point roundoff errors differ from fixed point in that the expected magnitude of the error due to a roundoff is proportioned to signal amplitude. The result of this is that, to perform a statistical analysis of floating point roundoff noise, one must assume a statistical model for the signal, as well as for the roundoff errors. Sandberg chooses instead to use a nonstatistical approach, and derives deterministic bounds on output noise-to-signal ratio. The difficulty with these bounds is that they apply to the situation where all errors add up in the worst possible way, and thus are quite pessimistic compared with results which actually occur in practice. Kaneko and Liu[24] analyze floating point roundoff noise via the statistical approach. The author, in collaboration with Oppenheim,[25] has applied the method of Kaneko and Liu to analyze some simple recursive filters, and has experimentally verified the model. This work, which also includes a comparison of the noise-to-signal ratios of fixed and floating point filters, will be discussed in Sec. II.

Oppenheim[26] has studied the effects of roundoff noise in simple recursive filters realized with block floating point arithmetic. These results, and a discussion of the comparison block floating point with fixed and floating point, will be reviewed in Sec. II-E.

Although most of the previous work on quantization noise in digital filters has treated recursive difference equation realizations, some study has been devoted to the problem of roundoff errors in the FFT. Welch[27] proposes and analyzes a model for roundoff noise in fixed point FFT. He also considers block floating point FFT, and obtains an upper bound on the output noise-to-signal ratio by considering the case where the array must be rescaled at each stage. Some experimental results, which are in reasonable agreement with the theory, are presented. In Sec. III-C, some additional work on fixed point FFT, and a noise analysis of block floating point FFT which takes signal statistics into account, will be presented.

Gentleman and Sande[28] studied the problem of roundoff errors in a floating point FFT. They chose to derive a deterministic bound on the output noise-to-signal ratio, rather than using a statistical approach. Their bound is rather pessimistic, and the author[29] has used a statistical approach to predict output noise-to-signal ratio in an FFT. This analysis, and experiments confirming the theory, will be presented in Sec. III-D. Gentleman and Sande also compared the accuracy of a floating point FFT to a floating point DFT via a direct summing of products, and conclude that the FFT is significantly more accurate. A method for arranging the summing of products, so that the "slow" Fourier transform is essentially as accurate as the FFT, will be discussed later in Sec. III-D.

For the present report, the techniques of statistical noise analysis, along with consideration of overflow constraints, will be used to derive expected noise-to-signal ratios for frequency sampling and FFT filters. Some work which will be reported relating to consideration of the overflow constraint in FFT filters has already been published by Oppenheim and the author.[30]

In general, the approach we shall use for the analysis of roundoff errors will be a statistical approach. There are effects of roundoff, important in some situations, which cannot be analyzed statistically. Such effects as the dead band effect[31] and limit cycles will, in general, not be discussed here.

C. ROUNDOFF ERROR MODELS

The fundamental source of errors incurred during the actual digital filtering computation is the error caused when the result of a multiplication or an addition must be rounded to a word length smaller than that needed to represent the exact result. To analyze this problem, one usually assumes a simple statistical model for the roundoff errors, and employs linear system noise theory to analyze their effect on the filter output. Different roundoff error models must be used for fixed and floating point arithmetic. The noise models cannot, in general, be verified theoretically, so that experiments must be used to test their validity.

We should mention that there is an alternative to using statistical roundoff error models; that is, one can derive deterministic bounds on the noise at the output of a filter. The chief difficulty with such bounds is that they are usually quite pessimistic, in comparison with the results of experiments. Our emphasis will be on the statistical approach.

In this section, statistical roundoff error models for both fixed and floating point computation will be presented and discussed. Some experiments will be described, where pertinent statistics of roundoff noise sequences have been measured, with a view toward testing the models. In general, the results are in accord with those of previous workers (see Sec. I-B-3).

Before proceeding to a discussion of the models, let us emphasize that roundoff in digital filtering is unavoidable, and cannot be prevented by increasing the register length by some number of bits. To illustrate this, consider the simple first-order recursion

$$y(nT) = ay(nT - T) + x(nT) \quad . \tag{I-4}$$

Consider fixed point computation, and assume that a and $y(nT - T)$ each contain t bits. Then their product will contain $2t$ bits. If the length, in bits, of this product is not reduced by rounding, it will grow by t bits for each successive iteration.

1. Fixed Point Roundoff Errors

In fixed point arithmetic, rounding errors need occur only when multiplications are performed. Fixed point additions are error-free provided no overflow occurs.

To discuss quantitatively the errors due to roundoff of products, we need to define a convention for the representation of numbers in fixed point. We shall assume a word length $t + 1$ bits, and shall consider our numbers to be fixed point fractions, consisting of t bits plus a sign, with the binary point just to the right of the sign bit. Thus, for a positive number, a one in the least-significant bit represents a numerical value of 2^{-t}. This convention will be used throughout this report. It implies no loss of generality, but merely assigns a scale of units to our fixed point numbers. The unit 2^{-t} will be referred to as the width of quantization.

When two of these fixed point fractions, for example, the constant a and the variable $y(nT - T)$ in (1-4), are multiplied together, the product will generally contain $2t$ bits plus a sign. But only t bits plus sign can be retained for use in future computation. The range of values which the resulting error can assume depends on exactly how the product is cut to single precision. If the product is rounded to the nearest single precision fraction, then the error will lie in the interval $[-(1/2)\ 2^{-t}, (1/2)\ 2^{-t}]$. But suppose that we truncate the double length product by simply dropping the lower order bits. Then, when the product is positive, the rounding error will lie in $(-2^{-t}, 0)$; and, when the product is negative, the error will lie in $(0, 2^{-t})$. Thus, for truncation, the error is signal dependent in that its sign depends on the sign of the resulting product.

For the most part, our analyses of roundoff noise will be based on rounding, rather than truncation, basically because it is analytically convenient to be able to make the assumption that the rounding noise is uncorrelated with the signal, but also because rounding generally causes less output noise than truncation. However, truncation can usually be implemented more simply and in less time, so that we shall often discuss this case and report on experiments measuring truncation noise.

Focusing our attention on rounding rather than truncation, let us mention two further points. First, the rounding process is not completely defined by saying that we round to the nearest quantization level, for we must define what to do when our double precision product lies exactly between two levels. This question arises, for example, in the case where we are multiplying our signal by the constant 1/2, for here our result will lie exactly between two quantization levels approximately half of the time. To be consistent with our assumption that rounding errors are signal independent, we shall assume that a random choice is made as to which of the equidistant quantization levels we shall choose. As we shall see later, there are situations where the mean-squared output noise in a digital filter (specifically, one involving FFT computation) will differ significantly, depending on whether, in this midway situation, we round up or down at random or choose, for example, to always round up. The second point is that we need not assume that the constant coefficients in the system, and the signals being processed, are retained to the same precision. For example, in a practical hardware implementation of a filter, we might want to use 12-bit coefficients and 16-bit data, rounding our 28-bit products to 16 bits.

Now let us define the statistical model we shall use for fixed point rounding errors. With our width of quantization taken as 2^{-t}, we assume the rounding errors to have a probability density function which is uniform in the interval $[-(1/2)\,2^{-t}, (1/2)\,2^{-t}]$. Thus, the variance of the rounding errors can easily be derived to be $(1/12)\,2^{-2t}$. The next key assumption we make is that the rounding error sequences are uncorrelated from sample to sample, that is, they represent digital white noise sources. Next, we assume that the roundoff noise is uncorrelated with the signal. Finally, we shall generally be considering computations where many multipliers are present and must be associated with a roundoff noise source; we shall assume all these noise sources to be mutually uncorrelated.

The basic model for rounding errors just defined will also be assumed to apply to noise due to A-D conversion. In addition, we shall assume that the A-D noise is uncorrelated with the roundoff noise.

Previous studies of these noise models, both theoretical and experimental, have been discussed in Sec. I-B-3. Some experiments, testing directly some of the assumptions of the models, will be reported below in Sec. I-C-3.

2. Floating Point Roundoff Errors

Floating point roundoff errors differ from fixed point roundoff errors in two key ways. First, the expected magnitude of a floating point roundoff error depends on the magnitude of the signal, since the numerical value of the last bit in the mantissa is determined by the exponent of the signal. Second, not only the results of multiplications, but also the results of additions must be rounded in floating point.

The following conventions will be used with respect to floating point computation. We assume a binary representation of floating point numbers in the form of (sign) $\cdot a \cdot 2^b$, where a and b each have a fixed number of bits. The number b, called the exponent, is an integer; the number a, called the mantissa, is a normalized fraction lying in the range 1/2 to 1. It will generally be assumed that enough bits are allotted to the exponent b so that no computed number will lie outside the permissible range. The notation $fl(\cdot)$ will denote the machine number resulting from the arithmetic operations specified by the parentheses. In general, the order in which the operations indicated inside the parentheses are carried out needs to be specified.

Suppose two floating point machine numbers x and y, each with a t-bit mantissa, are multiplied. Then, the exact product will have a 2t-bit mantissa, of which only the most-significant t bits can be retained. Depending on whether the mantissa is truncated or rounded, the sign of the error will or will not depend on the sign of the product xy. But, in either case, the numerical value of the error will be proportional to $2^{b(xy)}$, where b(xy) is the exponent of the product. Essentially, this means that the error is proportional to the product, a fact which can be summarized as[17]

$$fl(xy) = xy(1+\epsilon) \tag{I-5a}$$

where

$$|\epsilon| < 2^{-t} \quad . \tag{I-5b}$$

When two floating point machine numbers x and y are added, the mantissa of the exact sum will, in general, have more than t bits, even when the exponents of x and y are equal. The inexact computation can be expressed as

$$f\ell(x + y) = (x + y)(1 + \delta) \tag{I-6a}$$

where

$$|\delta| < 2^{-t} \quad . \tag{I-6b}$$

We now introduce a statistical model for the roundoff errors of floating point multiplication and addition. As in the fixed point case, the assumptions we make cannot be proven theoretically, and, in fact, can be demonstrated to be false for specific cases. Furthermore, our model for floating point computation cannot be derived from our model for fixed point computation. In fact, arguments can be constructed showing the two models to be inconsistent. However, there are two important justifications for the floating point roundoff model we shall use. First, the model is analytically convenient for use in predicting output noise in recursive filters and FFT's. Second, the predictions of the model have been found to be in reasonable accord with experimental noise measurements.

Our model, which applies to the case of rounding† rather than truncation, is as follows. We assume the roundoff variables ϵ and δ in (I-5) and (I-6) to be independent of the signal (the result of the multiplication or addition) and to have zero mean. Furthermore, where the multiplications and additions are used in iterative schemes, we assume these roundoff variables to be independent from iteration to iteration. We also assume that the roundoff variables associated with all the various multiplications and additions in a computation are independent of each other.

It remains to specify the first-order probability density function (pdf) of our roundoff variables. By analogy with our fixed point model, and in accordance with (I-5b) and (I-6b), we might be led to assume that ϵ and δ are each uniformly distributed in $(-2^{-t}, 2^{-t})$ with variances $\sigma_\epsilon^2 = \sigma_\delta^2 = (1/3)\ 2^{-2t}$. Experiments have shown that the actual pdf's are not quite uniform, and that the variances are slightly less than $(1/3)\ 2^{-2t}$. Also, the details of the pdf and exact value of the variances have been found to vary slightly with the signal being processed and the computation being performed. However, the assumption that σ_ϵ^2 and σ_δ^2 are proportional to 2^{-2t} has proved valid. Since our analysis of noise in digital filters generally involves only the variances σ_ϵ^2 and σ_δ^2, and not pdf's, our general approach will be to leave these variances as variables and use empirical numbers for them only when comparison of theory with experiment is to be carried out.

Typical empirical values for the roundoff variable variances are $\sigma_\epsilon^2 \approx \sigma_\delta^2 \approx 0.23 \times 2^{-2t}$. We should note that, for practical purposes, only rough estimates of these variances are needed, since even a 50-percent error in the variance represents less than one bit of difference in the predicted rms output noise.

One final point should be made about floating point roundoff models; that is, in order to perform a statistical analysis of noise in a floating point digital filter, one must assume a statistical model for the signal, as well as for the roundoff variables. To see this, consider the trivial digital filter

† As in the fixed point case, we assume that when a result lies equally between two quantization levels, a random choice is made as to whether to round up or down. This midway situation occurs frequently in floating point addition.

$$y(nT) = ax(nT) \tag{1-7}$$

where the output sequence is just a constant times the input sequence. Floating point computation would yield

$$fl\,[y(nT)] = ax(nT)\,[1 + \epsilon(n)] \tag{1-8}$$

with an error

$$e(n) = fl\,[y(nT)] - y(nT) = \epsilon(n)x(nT) \tag{I-9}$$

which is proportional to the signal x(nT). Given that we model x(nT) as a stationary random process with variance σ_x^2, we can find the error variance as

$$\sigma_e^2 = \sigma_\epsilon^2\sigma_x^2 \quad . \tag{1-10}$$

3. Experimental Verification of Roundoff Noise Models

Some experiments directed toward verifying the simple white noise model for roundoff errors have been carried out. The direct tests on the roundoff error models, to be described below, were carried out for the fixed point case. Experiments testing the models' predictions of output noise in digital filters, for both the floating and fixed point cases, will be described later.

Sample sequences of roundoff errors were obtained by multiplying signals by fixed coefficients, and saving the errors caused by rounding the results to single precision. The two types of signals used were sinusoidal signals generated by recursive difference equations with poles on the unit circle, and white noise sequences obtained from a random number generator. For these (and other) noise measurements, special signals for which the model is clearly invalid needed to be avoided. An example of such a signal is a sinusoid generated from a table lookup, where the sampling frequency is exactly divisible by the sine wave frequency. Here, the signal samples will repeat exactly each period. Thus, the roundoff error samples, obtained by multiplying the periodic signal by a fixed coefficient and rounding the results, will repeat with the same period, clearly contradicting the assumption of white roundoff noise.

For the first set of experiments, attention was directed to the case of fine quantization. The two types of signals used were generated as sequences of 18-bit words, were multiplied by 18-bit coefficients, and the results rounded to 18 bits.

First-order statistics (mean, variance, probability density function) of the error sequences were estimated experimentally. In all cases, as more data points were used, the mean converged quickly toward zero, and the variance approached the predicted value of $2^{-2t}/12$. Experimental histograms were taken to check the assumption of uniform first-order pdf. As more and more data were used, the results converged toward the assumed uniform density function.

The next step was to test the important assumption that the roundoff error sequences are uncorrelated from sample to sample. To make this investigation quantitative, a technique presented in Grenander and Rosenblatt (Ref. 18, pp. 97-101) was used to obtain theoretically the probability density function for the first correlation coefficient (the autocorrelation function for a lag of one sample, divided by the autocorrelation for a lag of zero), for the case of white sequence. The approximate result was that for a white sequence of N samples, the first correlation coefficient ρ_1 would have a normal distribution with mean of $-(1/N)$, and variance $1/N$. By using this result, a significance test was performed on measured values of ρ_1 for roundoff error sequences. We tested 3000 error sequences of length N = 512, obtained from rounding of sinusoidal signals of 500 different frequencies, multiplied by 6 different coefficients. At the 1-percent significance level, 1.1 percent of the measured values of ρ_1 failed the significance test, and at the 5-percent significance level, 5.83 percent of the tests failed. These results are in quite reasonable accord with the white noise model for roundoff error. In addition to checking when significance levels were exceeded, an experimental histogram was obtained for the 3000 measured correlation coefficients. The distribution followed reasonably well the predicted normal distribution.

Similar experiments were then performed for coarser quantization, that is, rounding was done to fewer and fewer bits, and correlation coefficients of error sequences were measured. For roundoff of the sinusoidal signals used, even down to 4 bits, the assumption of uncorrelated samples was not contradicted for the significance test just discussed. Measurements of the second, third, fourth and fifth correlation coefficients also tended to agree with the assumption of uncorrelated samples. These results are in accord with experiments reported by Sornmoonpin.[16]

Finally, spectral measurements were performed on roundoff error sequences obtained by multiplying 18-bit sine waves by 18-bit coefficients and rounding the results to 18 bits. The most successful measurement technique used was to section the noise sequence into sections 2048 points long, compute periodograms of the individual sections via the FFT, smooth each periodogram via a moving average smoothing, and then average the results over all the smoothed periodograms. We found that if enough data were used, the measured spectra would begin to look white, but that the amount of data needed was quite large. In one example, a smoothing window of spectral width 1/16 the sampling frequency was used on each periodogram, so that actually only 16 independent points on the spectrum were obtained. Still, 24 sections, or about 48,000 data points, had to be used before the mean-squared deviation of the spectrum from its average was less than 5 percent of the average. Such results are actually in reasonable accord with the kind of convergence predicted by spectral estimation theory. Since spectral measurements on white noise should converge faster than measurements on colored noise, the implication is that in order to make accurate spectral measurements on output noise in digital filters, one would have to find ways to obtain very long noise sequences and be willing to expend much computation time for their analysis.

D. SUMMARY

Before proceeding to the body of the report, let us now briefly outline the order of topics to be studied.

Section II will concentrate on quantization effects in simple (first- and second-order) recursive filters. The analysis of coefficient quantization will be reviewed for the fixed point case, and carried over to the floating point case. The work of others analyzing roundoff noise for the fixed point case will be reviewed and considered jointly with the overflow constraint to obtain output noise-to-signal ratios. The use of a clamp to counteract the problem of overflow in fixed point recursions will be investigated. Noise-to-signal ratios for the block floating and floating point cases will be derived, and experimental verification presented. A comparison of the three different types of arithmetic, on the basis of differing noise-to-signal ratios, will be presented for these first- and second-order recursive filters.

In Sec. III, quantization effects will be studied for the other basic algorithm of digital filtering – the FFT. The effect of coefficient quantization will be studied both theoretically and experimentally for the fixed and floating point cases. With respect to roundoff errors, noise to-signal ratio analyses for fixed, block floating, and floating point will be derived, and experimental results will be presented for all three types of arithmetic. A comparison of quantization effects in the three types of arithmetic will be presented.

Section IV is devoted to frequency sampling filters. The coefficient accuracy problem is investigated in detail for the fixed point case, and experimental results are presented for a specific filter example. Floating point coefficient errors are discussed. Section III's fixed point noise analysis of digital resonators is applied to derive noise-to-signal ratios for the frequency sampling filter. Alternate realizations for the digital resonators are compared. On the basis of the coefficient and roundoff noise analysis, and the overflow constraint,. register length requirements for the realization of a fixed point frequency sampling filter are set forth. Comparison with the floating point case is discussed.

In Sec. V, quantization effects in FFT filters are analyzed, and a comparison with alternate realizations of the same filter via frequency sampling is included. Applying the results of Sec. III, we examine the effect of coefficient errors in an FFT filter. Experimental measurements of the actual error in the filter's frequency response due to coefficient quantization are reported. An upper bound on the output of a circular convolution is derived, which assists one in arranging scaling of the array to prevent overflow in a fixed point FFT filter. For the fixed point case, the output noise-to-signal ratio is analyzed and the results applied specifically to the same filter example considered in Sec. IV in the frequency sampling form. A comparison of the FFT vs the frequency sampling realization is discussed.

Finally, in Sec. VI, the results of the report are discussed, and some problems for further study are proposed.

II. SIMPLE RECURSIVE FILTERS

This section will be devoted to a study of quantization effects in first- and second-order recursive digital filters. We study these simple recursions in much detail because they are the basic building blocks from which complicated recursive digital filters should be constructed. First, we shall investigate the coefficient quantization problem, then consider jointly the problems of roundoff noise and overflow in order to predict output noise-to-signal ratios for these simple filters. Fixed, block floating, and floating point arithmetic all will be considered and compared on the basis of differing quantization effects.

A. COEFFICIENT QUANTIZATION – FIXED POINT

Here we review the analysis of the effects of coefficient quantization in simple recursive filters. The analysis is quite similar to that in Refs. 2 and 9. Attention will be mainly directed toward the effect of coefficient quantization on the pole positions of the filters.

For a first-order difference equation

$$y(nT) = ay(nT - T) + x(nT) \tag{II-1}$$

there is one coefficient a, and the pole position is equal to the value of this coefficient. Thus, if rounding causes the filter to be realized with $a + \Delta a$, then the error in the pole position is Δa. For fixed point coefficients of $t + 1$ bits, the pole positions which can be achieved are spaced uniformly along the real axis in the z-plane with separation 2^{-t}. The error in pole position is bounded by $\frac{1}{2}2^{-t}$, the maximum error in a fixed point coefficient.

The situation is more complicated for a second-order equation such as

$$y(nT) = a_1y(nT - T) + a_2y(nT - 2T) + x(nT) \tag{II-2}$$

which will be assumed to have complex conjugate poles at $r\ e^{\pm j\Theta}$, where the relationship between coefficients and pole positions is specified by

$$a_1 = 2r \cos\Theta \quad \text{and} \quad a_2 = -r^2 \tag{II-3}$$

or equivalently

$$r = \sqrt{-a_2} \quad \text{and} \quad \Theta = \cos^{-1}(a_1/2\sqrt{-a_2}) \quad . \tag{II-4}$$

Assuming small quantization errors, we can approximate the sensitivity of pole positions to coefficient errors by differentiating r and Θ with respect to a_1 and a_2. The result is

$$\Delta r \approx -\Delta a_2/2r \quad , \quad \Delta\Theta \approx -\Delta a_2/2r \tan\Theta - \Delta a_1/2r \sin\Theta \quad . \tag{II-5}$$

The error in radius, for the interesting case of r near unity, is not very different from the case of the first-order equation; but the error in Θ, for Θ near 0, can be quite large. This can be quite troublesome, for example, in low-pass frequency sampling filters, which require accurately tuned resonators with resonant frequencies which are a small fraction of the sampling rate.

To see this problem from another viewpoint, consider that the quantization of a_1 and a_2 defines a grid of allowable pole positions in the z-plane.† From (II-3) one can deduce that the grid is defined by intersections of uniformly spaced straight lines parallel to the imaginary axis, with

† J. Evans, private communication.

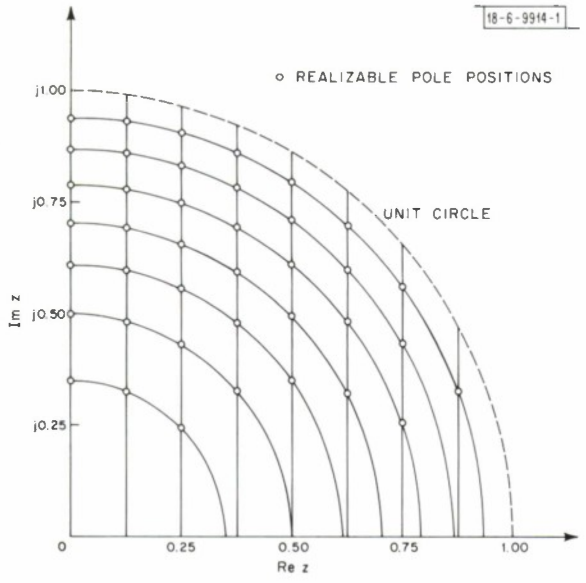


Fig. 1. Grid of ollowoble pole positions – direct form.

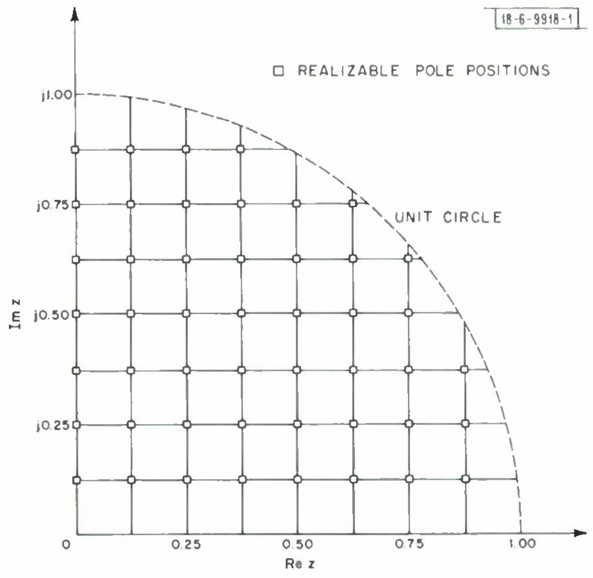


Fig. 2. Grid of ollowoble pole positions – coupled form.

circles centered on the origin and becoming more closely spaced as r approaches 1. Such a grid is pictured in Fig. 1. For small Θ and r near 1, we can see that the circles and straight lines tend to become parallel, and quantization of the angles of the poles becomes very coarse.

A second-order transfer function in which the pole positions are less sensitive to parameter errors can be realized via the pair of coupled first-order equations,

$$y_1(nT) = a_1 y_1(nT - T) - a_2 y_2(nT - T) + b_1 x(nT)$$

$$y_2(nT) = a_2 y_1(nT - T) + a_1 y_2(nT - T) + b_2 x(nT) \quad . \qquad \text{(II-6)}$$

The relationship between coefficients and pole positions is given by

$$a_1 = r \cos\Theta \quad , \quad a_2 = r \sin\Theta \quad . \qquad \text{(II-7)}$$

Expressions for approximate errors in pole positions can be obtained as

$$\Delta r \approx (\cos\Theta)\, \Delta a_1 + (\sin\Theta)\, \Delta a_2$$

$$\Delta\Theta \approx \left(\frac{\cos\Theta}{r}\right) \Delta a_2 - \left(\frac{\sin\Theta}{r}\right) \Delta a_1 \quad . \qquad \text{(II-8)}$$

These errors in pole positions are quite different than those given in (II-5), and for Θ small, the error in Θ is substantially less for the realization of (II-6) than for (II-2). Of course, (II-6) requires substantially more computation than does (II-2), and thus should be resorted to only when quantization of parameters is a problem. (Actually, the coupled form, for certain cases, yields an improved noise-to-signal ratio over the direct form, and might be resorted to for this reason also.) We shall see later (in Sec. IV) that for the case of a low-pass frequency sampling filter, coupled form resonators yield distinct improvements over the performance obtained using direct form resonators.

It is interesting to examine the grid of allowable pole positions for the coupled form. One can deduce from (II-7) that this is a uniform rectangular grid as depicted in Fig. 2. In contrast to Fig. 1, quantization of the angles of the poles does not become especially coarse for small Θ, but is essentially equally fine for all Θ.

B. COEFFICIENT QUANTIZATION – FLOATING POINT

The equations of the preceding section for the relationships between coefficients and pole positions, and for the sensitivity of pole position to small changes in the coefficients, are valid for the floating point case as well as for the fixed point case. The only distinction is that the actual floating point coefficients are not uniformly quantized over their full range of values. This fact can be included approximately in the sensitivity formulas (II-5) and (II-8) by expressing the errors in the coefficients as percentage errors, proportional to the coefficients themselves. Additional insight can be gained by considering the structure of the grid of allowable pole positions, for the floating point case.

For a floating point first-order filter as in (II-1), the interesting range of parameters is generally restricted to the range $\frac{1}{2} < a < 1$. For this range, the allowable pole positions are uniformly spaced along the real axis, as in the fixed point case. The spacing for the floating point case is 2^{-t}, where t now represents the number of bits in the mantissa.

In the next few paragraphs, our comparisons between floating and fixed point will be based on the implied assumption that the number of bits in the floating point mantissa equals the number of bits (not including sign) in the fixed point word. It should be pointed out that this implies a longer total register length for floating point, due to the extra bits needed for the exponent. However, our intention is to indicate qualitatively the differences in how a roundoff error affects pole position in the two types of arithmetic, and a quantitative comparison on the basis of equal register lengths is not attempted.

Let us consider the direct form second-order filter (II-2). We assume roundoff errors in the coefficients of the form

$$\Delta a_1 = \epsilon_1 a_1 \quad , \quad \Delta a_2 = \epsilon_2 a_2 \tag{II-9}$$

where

$$|\epsilon_1| < 2^{-t} \quad , \quad |\epsilon_2| < 2^{-t} \quad . \tag{II-10}$$

Substituting (II-9) and (II-3) in (II-5), we obtain sensitivity equations for the floating point case as

$$\Delta r \approx \epsilon_2(r/2) \quad , \quad \Delta\Theta \approx \epsilon_2(r/2 \tan\Theta) - \epsilon_1/\tan\Theta \quad . \tag{II-11}$$

For the interesting case of r near unity, the sensitivity of the pole radius is essentially the same as for fixed point. If, in addition, Θ is small, the error $\Delta\Theta$ is proportional to $1/\Theta$ as in the fixed point case. Thus, for high-gain filters where the ratio of resonant frequency to sampling rate is fairly low, the coefficient quantization problems for floating and fixed point are essentially the same. For filters where Θ approaches $\pi/2$, $\Delta\Theta$ becomes very much smaller than in the fixed point case since the coefficient $a_1 = 2r\cos\Theta$ becomes small, causing errors in this coefficient to become proportionately small.

As in the fixed point case, the grid of allowable pole positions for the direct form is defined by intersections of straight lines parallel to the imaginary axis, with circles centered on the origin. When r is near unity, the relative spacing of the circles is essentially the same as in the fixed point case depicted in Fig. 1. For $r\cos\Theta = \mathrm{Re}\, z$ in the range 1/2 to 1, corresponding to the case of reasonably small Θ, the relative spacing of the straight lines is also essentially the same as in the fixed point case. For $r\cos\Theta$ much less than 1/2 (that is, where Θ is near $\pi/2$), the straight lines are spaced much more closely than in the fixed point case.

Now let us consider the coupled form resonator of (II-6). Substituting (II-9) and (II-7) in (II-8), we obtain the sensitivity formulas

$$\Delta r \approx \epsilon_1 r\cos^2\Theta + \epsilon_2 r\sin^2\Theta \tag{II-12a}$$

$$\Delta\Theta \approx (\epsilon_2 - \epsilon_1)\left(\frac{\sin 2\Theta}{2}\right) \quad . \tag{II-12b}$$

The error in r is slightly, but not significantly, different from the fixed point case. The sensitivity in Θ is never greater than in the fixed point case, and for Θ near 0 or $\pi/2$, the factor $\sin 2\Theta$ in (II-12b) implies that the error in Θ will be much smaller than in the fixed point case.

The grid of allowable pole positions, as in the fixed point case, is rectangular, but the lines are not uniformly spaced. For the area of the z-plane (considering only the first quadrant), where

$$\frac{1}{2} < \mathrm{Re}\, z = a_1 = r\cos\Theta < 1$$

$$\frac{1}{2} < \mathrm{Im}\, z = a_2 = r\sin\Theta < 1 \tag{II-13}$$

the lines are spaced uniformly with separation 2^{-t}, as in fixed point. But for small values of Re z or Im z, corresponding to small values of the coefficients a_1 or a_2, and to resonant angles near 0 or $\pi/2$, the grid becomes much finer. As indicated in (II-9), when the coefficient a_1 or a_2 becomes small, the error due to quantizing this coefficient becomes proportionately small.

C. ROUNDOFF NOISE AND OVERFLOW CONSTRAINT – FIXED POINT

In this section, we analyze the effects of roundoff noise in first- and second-order filters implemented using fixed point arithmetic. The derivations of output noise variance are a review of the work of others.[20-22] Experimental results reported by these previous workers have generally supported the theoretical results for output noise variance. We shall discuss how the overflow constraint, considered jointly with these noise formulas, determines output noise-to-signal ratio, and shall derive formulas for output noise-to-signal ratio, for the case of a flat signal spectrum. An alternative to complete prevention of overflow, the insertion of a saturating device or clamp in the feedback loop, will be discussed. Some experimental observations of the distortions introduced by such a device are reported.

Although we shall not explicitly analyze the effects of A-D noise, it should be apparent that this noise could be easily included, using the same approach as for roundoff noise. However, our focus of attention is on the roundoff noise, since this is the noise introduced explicitly by the digital computation. With this viewpoint, we can compare the accuracies of various algorithms which might be used to process signals which have been already digitized.

1. General Formulas for Noise Variance

Our general approach to the analysis of roundoff noise in fixed point digital filters is as follows. We model all the noisy multipliers as noiseless multipliers combined with additive white noise sources. Since, according to the model of Sec. I-C-1, all the noise sources are independent of each other and of the signal, we can determine mean, variance, or spectrum of the output noise simply by adding the effects of all the individual noise sources. To determine the effect of any individual noise source, we find (from inspection of the circuit diagram of the filter, or otherwise) a transfer function between the noise source and the filter output. We then use linear system noise theory to derive the statistics of the output noise produced by this particular noise source. Thus, the basic situation we must analyze is as depicted in Fig. 3. Given white noise $\epsilon(nT)$ as input to an arbitrary transfer function H(z), what are the statistics (mean, variance, etc.) of the output e(nT)?

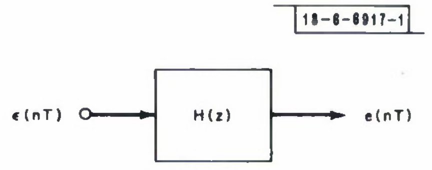


Fig. 3. White noise exciting a linear digital filter.

We can conveniently examine this model via the convolution sum. Thus,

$$e(nT) = \sum_{m=0}^{n} h(mT)\,\epsilon(nT - mT) \qquad \text{(II-14)}$$

where h(mT), the inverse z transform of H(z), is the impulse response of the filter. The input noise $\epsilon(nT)$ is presumed to be zero for $n < 0$, and the system is initially at rest. The roundoff error sequence $\epsilon(nT)$ is assumed to have zero mean, variance

$$\sigma_o^2 = 2^{-2t}/12 \qquad \text{(II-15)}$$

and be independent from sample to sample. Thus, e(nT) is a weighted sum of independent random variables. Its mean is zero, and its variance is

$$\sigma_e^2 = \sigma_o^2 \sum_{m=0}^{n} h^2(mT) \quad . \tag{II-16}$$

The power spectrum of e(nT) is simply the squared magnitude of the transfer function $H(e^{j\omega T})$.

For a stable system with all its poles inside the unit circle, the right side of (II-16) converges, and the steady-state variance of e(nT) can be obtained by letting $n \rightarrow \infty$. We shall usually work with this steady-state result,

$$\sigma_e^2 = \sigma_o^2 \sum_{m=0}^{\infty} h^2(mT) \quad . \tag{II-17}$$

Often, instead of directly evaluating the sum in (II-17), it is simpler to use the identity

$$\sum_{m=0}^{\infty} h^2(mT) = \frac{1}{2\pi j} \oint H(z)\, H(1/z)\, z^{-1} dz \tag{II-18}$$

and employ the method of residues to perform the contour integration.

2. Noise-to-Signal Ratio – First-Order Case

Let us apply the technique just discussed to obtain the output noise variance for a first-order system, as represented in Fig. 4, or by the equation

$$y(n) = ay(n-1) + \epsilon(n) + x(n) \quad . \tag{II-19}$$

In (II-19), and in the remainder of the report, we assume for convenience that our units of time have been chosen so that the sampling interval T = 1. This implies no loss of generality.

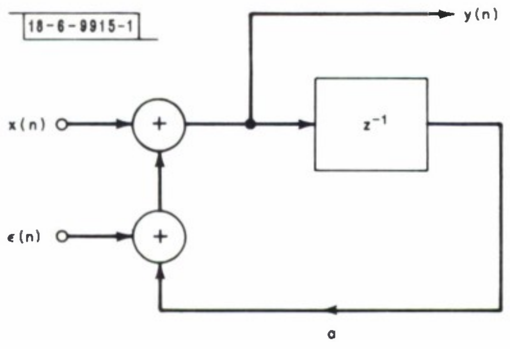


Fig. 4. Noisy first-order filter (fixed point).

The noise source ϵ(n) enters the system at the same point as does the input x(n). The system function H(z) is given by $1/(1 - az^{-1})$, and $h(m) = a^m$. Using (II-17), the steady-state output noise variance can be easily computed as

$$\sigma_e^2 = \sigma_o^2 \sum_{m=0}^{\infty} a^{2m} = \frac{\sigma_o^2}{(1-a^2)} \quad . \tag{II-20a}$$

For the case of a high-gain, narrowband filter, with $a = 1 - \delta$, and δ small, (II-20a) tends toward

$$\sigma_e^2 = \frac{\sigma_o^2}{2\delta} \quad . \tag{II-20b}$$

The implications of these noise formulas are tricky. The mean-squared value of the noise clearly increases as a approaches unity. But this increase in filter gain also means that the output signal power will increase. For a signal with flat spectrum, the signal input goes through the same power gain as does the noise source, so that the output signal-to-noise ratio would seem not to change as the pole of the filter moves toward the unit circle. For a low-frequency signal, it would seem that the output signal-to-noise ratio would increase as the pole moves toward $z = 1$. However, these results are misleading, since they do not take into account the fact that with a finite word length, the input signal must be kept small enough so that it does not cause overflow in the computation. Thus, the obtainable signal-to-noise ratio decreases as a approaches unity.

Although the formula (II-20a) for output noise variance does not depend on the signal being processed, it is clear that the output noise-to-signal ratio in the filter does depend on the type of input signal. However, we would like to obtain quantitative results for noise-to-signal ratio, and in order to do so must assume a specific model for the signal. We choose to analyze the situation where the signal is white, that is a sequence of zero mean, uncorrelated random variables. This is a sort of intermediate situation, between the extreme cases of a signal whose spectrum is completely in the filter passband, and one where spectrum is out of band. It is a case which lends itself fairly readily to analysis, and where the results can be compared easily with results to be derived later for floating and block floating point filter realizations. The caution must be stated that for cases where the signal spectrum is not broad, our results may be misleading. To illustrate the dependence of noise-to-signal ratio on signal spectrum, we shall derive also, for a first-order filter, the noise-to-signal ratio for sinusoidal input. It should be clear that the techniques we use can be straightforwardly applied to obtain noise-to-signal ratios for any given signal spectrum.

Now let us proceed to mathematically apply the overflow constraint to our first-order filter, and, on the basis of white signal statistics, derive a formula for the mean-squared output noise-to-signal ratio. According to our convention that fixed point words are considered as signed fractions, the requirement for no overflow is that no output of the filter exceed unity in value. Thus, we require that

$$|y(n)| < 1 \quad , \quad \text{all } n \quad . \tag{II-21}$$

In order to translate this constraint into a bound on the input sequence, we observe that the maximum possible output value can be expressed in terms of the maximum allowed input value as

$$\max\left(|y(n)|\right) = \max\left(|x(n)|\right) \sum_{n=0}^{\infty} |h(n)| \quad . \tag{II-22}$$

From (II-21) and (II-22), we deduce that to guarantee no overflows, $x(n)$ must be restricted to the range

$$\frac{-1}{\sum_{n=0}^{\infty} |h(n)|} < x(n) < \frac{1}{\sum_{n=0}^{\infty} |h(n)|} \quad . \tag{II-23}$$

We shall assume x(n) white and uniformly distributed between the limits in (II-23). Thus, the input signal variance for our first-order filter is

$$\sigma_x^2 = \frac{1}{3\left(\sum_{n=0}^{\infty} |h(n)|\right)^2} = \frac{(1-a)^2}{3} \quad . \tag{II-24}$$

The variance of the output signal is

$$\sigma_y^2 = \frac{1}{(1-a^2)}\sigma_x^2 = \frac{(1-a)}{3(1+a)} \tag{II-25}$$

and dividing (II-20a) by (II-25) we obtain the output noise-to-signal ratio as

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{3\sigma_o^2}{(1-a)^2} \quad . \tag{II-26a}$$

Writing $a = 1 - \delta$ and substituting (II-15) in (II-25), we obtain

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{1}{4}\frac{2^{-2t}}{\delta^2} \quad . \tag{II-26b}$$

Thus, the mean-squared output noise-to-signal ratio is proportional to the reciprocal of the square of the distance of the pole from the unit circle.

We should re-emphasize that for the case we are considering (white input signal), the signal and noise go through the same gain, independent of where the pole is. It is the overflow constraint restricting the input signal level which causes noise-to-signal ratio to change with filter gain.

For contrast, we shall now obtain σ_e^2/σ_y^2 for the case of a sinusoidal signal $x(n) = A\cos(\omega_o n + \varphi)$, where φ is uniformly distributed in $(-\pi, \pi)$. To satisfy (II-23), we set

$$A = 1 - a$$

so that

$$\sigma_x^2 = \frac{(1-a)^2}{2} \quad .$$

Now

$$\sigma_y^2 = \left|H\left(e^{j\omega_o}\right)\right|^2 \sigma_x^2 = \frac{\sigma_x^2}{1 + a^2 - 2a\cos\omega_o}$$

or

$$\sigma_y^2 = \frac{(1-a)^2}{2(1 + a^2 - 2a\cos\omega_o)}$$

and dividing this result into (II-20a) we obtain the ratio

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{(2^{-2t})}{6(1-a)^2} \frac{(1 + a^2 - 2a \cos \omega_o)}{(1-a^2)} \quad . \qquad \text{(II-26c)}$$

If we let $\omega_o \rightarrow 0$, so that the signal goes through the maximum filter gain, this approaches

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2t}}{6(1-a)^2} \frac{(1-a)^2}{(1-a^2)}$$

which, for $a = 1 - \delta$ and δ small, tends toward

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2t}}{12\delta} \quad . \qquad \text{(II-26d)}$$

Thus, if we excite our high-gain first-order filter with a low-frequency sinusoid, σ_e^2/σ_y^2 is proportional to $1/\delta$ rather than to $1/\delta^2$ as in (II-26b). For a sinusoidal input of low enough frequency, the signal goes through more gain than does the roundoff noise. This is in contrast to the case of white input, where the signal and noise go through the same gain.

3. Noise-to-Signal Ratios – Second-Order Case

We now proceed to derive formulas for output noise variance for three alternate realizations of the same two-pole, one-zero transfer function. Output noise-to-signal ratios will be obtained for the case of a white input signal.

The transfer function of concern is

$$H(z) = \frac{1 - r \cos \Theta \, z^{-1}}{1 - 2r \cos \Theta \, z^{-1} + r^2 z^{-2}} \qquad \text{(II-27)}$$

having a complex pole pair at $z = re^{\pm j\Theta}$, and a zero on the real axis at $z = r \cos \Theta$. The system behaves as a digital resonator with resonant frequency determined by the angle Θ and the sampling rate.

The first realization to be considered is the direct form, consisting of the single second-order difference equation

$$y(n) = 2r \cos \Theta \, y(n-1) - r^2 y(n-2) + x(n) - r \cos \Theta \, x(n-1) \quad . \qquad \text{(II-28)}$$

Inspection of a circuit diagram[2, 21] for (II-28) would show that the noise sources due to the three multiplications pass through only the poles of the filter. The resulting output noise variance is

$$\sigma_e^2 = 3G_1 \sigma_o^2 \qquad \text{(II-29a)}$$

where

$$G_1 = \left(\frac{1 + r^2}{1 - r^2}\right)\left(\frac{1}{r^4 + 1 - 4r^2 \cos^2 \Theta + 2r^2}\right) \quad . \qquad \text{(II-29b)}$$

For the high-gain case, where $r = 1 - \delta$ and δ is small, this becomes

$$\sigma_e^2 = \frac{3\sigma_o^2}{4\delta \sin^2 \Theta} \quad . \tag{II-30}$$

We have assumed that rounding of the products is done before performing the needed sums to carry out the recursion. If we were to accumulate the sums in, say, a double precision accumulator, before rounding, the output noise would be somewhat reduced.

The "canonic form" realization for our system is represented by the equations

$$\begin{aligned} u(n) &= x(n) + 2r \cos\Theta \; u(n-1) - r^2 u(n-2) \\ y(n) &= u(n) - r \cos\Theta \; u(n-1) \quad . \end{aligned} \tag{II-31}$$

The essential difference between the canonic and direct forms is that in the canonic form the signal passes first through the poles of the filter and then through the zero, while the reverse is true for the direct form. In the canonic form, the two important noise sources in the system [those introduced in computing u(n)] are filtered by the zero, as well as by the poles. The output noise variance is

$$\sigma_e^2 = 2G_2\sigma_o^2 \tag{II-32a}$$

where

$$G_2 = \frac{1}{1-r^2}\left[1 - \frac{r^2 \sin^2\Theta\,(1+r^2)}{r^4 + 1 - 4r^2\cos^2\Theta + 2r^2}\right] \tag{II-32b}$$

or, for r close to unity,

$$\sigma_e^2 = \frac{\sigma_o^2}{2\delta} \quad . \tag{II-33}$$

Comparing (II-30) and (II-33), we see that the direct form leads to increased noise for low resonant frequencies whereas the canonic form does not. We might be led to consider the canonic form superior on this basis, but one must consider signal levels as well as noise levels in comparing alternate forms. Since, in the canonic form, the signal passes first through the poles and then the zero, the input signal will generally have to be smaller than in the direct form, in order to prevent overflow. A noise-to-signal ratio analysis, as given below, yields essentially the same results for the two forms.

To prevent overflow for the direct form (II-28), we require that $|y(n)| < 1$, or since the overall impulse response is

$$h(n) = r^n \cos(n\Theta) \tag{II-34}$$

that

$$|x(n)| < \frac{1}{\sum_{n=0}^{\infty} r^n\,|\cos(n\Theta)|} \quad . \tag{II-35}$$

Assuming x(n) white and uniformly distributed according to the constraint (II-35), the output noise-to-signal ratio for the direct form is

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{9G_1\sigma_o^2}{G_2} \frac{1}{\left[\sum_{n=0}^{\infty} r^n |\cos(n\Theta)|\right]^2} \quad . \tag{II-36}$$

We can approximately bracket the expression

$$\sum_{n=0}^{\infty} r^n |\cos(n\Theta)|$$

by noting that an upper bound is

$$\sum_{n=0}^{\infty} r^n = \frac{1}{1-r} \quad .$$

A lower bound is obtained by observing that the sum of the absolute values of an impulse response is the maximum attainable output value from a filter if the maximum input value is unity. The maximum output of our second-order system at resonance (which can be found by evaluating H(z) in (II-27) at $z = e^{j\Theta}$) thus provides a lower bound on the sum of the absolute values of the impulse response. For the high-gain case, this resonant gain is approximately $1/2\delta$. Thus, we shall consider

$$\frac{1}{2\delta} \leqslant \sum_{n=0}^{\infty} r^n |\cos(n\Theta)| \leqslant \frac{1}{\delta} \quad . \tag{II-37}$$

Using these approximations in (II-36), and noting that $(G_1/G_2) \approx 1/\sin^2\Theta$ for the high-gain case, we have for the direct form that

$$\frac{9\sigma_o^2}{4\delta^2 \sin^2\Theta} \leqslant \frac{\sigma_e^2}{\sigma_y^2} \leqslant \frac{9\sigma_o^2}{\delta^2 \sin^2\Theta} \quad \text{(direct form)} \quad . \tag{II-38}$$

To prevent overflow in the canonic form (II-31), we require $|u(n)| < 1$,† or since the impulse response of the filter (poles only) between x(n) and u(n) is

$$h(n) = r^n \frac{\sin[(n+1)\Theta]}{\sin\Theta} \tag{II-39}$$

we require

$$|x(n)| < \frac{1}{\frac{1}{\sin\Theta} \sum_{n=0}^{\infty} r^n |\sin(n+1)\Theta|} \quad . \tag{II-40}$$

This is a lower bound on the input signal than in (II-35). Using approximations similar to the above, the output noise-to-signal ratio for the canonic form may be bracketed as

$$\frac{6\sigma_o^2}{4\delta^2 \sin^2\Theta} \leqslant \frac{\sigma_e^2}{\sigma_y^2} \leqslant \frac{6\sigma_o^2}{\delta^2 \sin^2\Theta} \quad \text{(canonic form)} \quad . \tag{II-41}$$

†Once u(n) is computed, it must be ottenuated by at least $1/(1 + |r \cos\Theta|)$ so thot y(n) does not overflow. This ottenuation hos only minor effects on the output noise-to-signal rotio, and will be neglected.

This predicts essentially the same behavior of noise-to-signal ratio, as a function of bandwidth and resonant frequency, as the direct form result (II-38).

Now we consider a third realization of our filter, via the coupled first-order equations

$$y(n) = r \cos\Theta \; y(n-1) - r \sin\Theta \; u(n-1) + x(n)$$
$$u(n) = r \sin\Theta \; y(n-1) + r \cos\Theta \; u(n-1) \quad . \tag{II-42}$$

Assuming (as we have throughout this discussion) that we round after each multiplication, the output noise variance for r near unity is

$$\sigma_e^2 = \frac{\sigma_o^2}{\delta} \tag{II-43}$$

the same (except for a factor of 2) as the canonic form. But if we consider also the overflow constraint, we shall find that at low resonant frequencies the coupled form noise-to-signal ratio is superior to the canonic and direct forms. This is because the noise is less than in the direct form, and the transfer function $U(z)/X(z)$ relating the intermediate variable $u(n)$ to the input has essentially the same gain as that relating $y(n)$ to the input, unlike for the canonic form where $U(z)/X(z)$ becomes large for low resonant frequencies. Using approximations similar to those used in obtaining (II-38) and (II-41), we can bracket the noise-to-signal ratio as

$$\frac{12}{4} \frac{\sigma_o^2}{\delta^2} \leqslant \frac{\sigma_e^2}{\sigma_y^2} \leqslant \frac{12\sigma_o^2}{\delta^2} \quad \text{(coupled form)} \quad . \tag{II-44}$$

Unlike in the direct and canonic forms, σ_e^2/σ_y^2 does not increase as the resonant frequency of the filter is decreased.

4. Clamping

As we have seen, the requirement of no overflows severely limits the attainable signal-to-noise ratio at the output of a fixed point recursive filter. An alternative to completely preventing overflow is to insert a clamp, or saturating element, in the feedback loop of the filter.

To illustrate the use of a clamp, consider the digital resonator

$$y(n) = a_1 y(n-1) + a_2 y(n-2) + x(n) \quad .$$

When the magnitude of the sum on the right-hand side of this equation becomes too large for the fixed point computer word (greater than 1), the output is made to assume the largest fixed value which the word length allows, rather than permitting an overflow and abrupt change of sign of the output. This clamped result is then used in computing the next output.

Clearly, insertion of a clamp would permit larger input signal levels to be used, and thus increase the ratio of output signal to noise. But this apparent improvement must be traded off against the distortion introduced by the clamp. For example, in situations where overflows would occur only rarely, a clamp might be a useful device, either to improve the output signal-to-noise ratio or to permit a system to be realized with shorter word length.

The theoretical analysis of a clamped filter is quite complicated, since the system is nonlinear, with the nonlinearity in a feedback loop. Therefore, a series of experiments were performed to investigate the effects of clamping in both first- and second-order filters.

In one series of experiments, the inputs used were sine waves (small enough to cause no overflow) with large, overflow-producing pulses added. With the clamp, the effect of the pulses died out in the transient response time of the filter, and steady-state filter behavior was restored. If the clamp was removed and an overflow allowed to occur, it was found that a sequence of many additional overflows were caused, and many time constants of the system would pass before steady-state behavior was restored.

In another set of experiments, the inputs used were sine waves large enough to cause overflow. When overflow was allowed to occur, the resulting outputs in no way resembled the outputs that would have occurred with no overflow. When a clamp was used, the resulting outputs, for a variety of filter parameters and input frequencies, resembled clipped sine waves and retained the frequency of the input. Such distortion might not be intolerable in a chain of resonators, for example, where the harmonic distortion might be filtered out by other resonators in the chain.

These experiments indicated that a clamp would be a useful device, e.g., in a hardware digital filter with severely limited word length. One would expect that a study of the effects of clamping in some practical system, such as a digital vocoder, might well yield some savings in register length requirements.

D. ROUNDOFF NOISE – FLOATING POINT

In this section, we analyze the effects of roundoff noise in first- and second-order, floating point digital filters. Formulas for output noise-to-signal ratio are obtained, which later will be compared with the corresponding results for fixed and block floating point arithmetic. Our statistical method of analysis follows that of Kaneko and Liu.[24] Some of the results, both theoretical and experimental, have been previously reported by Weinstein and Oppenheim.[25]

We shall employ the model for floating point roundoff errors which has been presented in Sec. I-C-2, assuming rounding rather than truncation. With respect to the overflow question, we shall assume that enough bits are allotted to the exponent so that no overflows occur. Later, in comparing floating and fixed point, we shall consider how many bits are actually needed in the exponent to satisfy this assumption.

In our analyses, several of the formulas given in the preceding section on fixed point arithmetic will be found helpful also for the floating point case.

1. First-Order Case

For a first-order filter of the form

$$y(n) = ay(n-1) + x(n) \tag{II-45}$$

the actual output computed with floating point arithmetic can be expressed as

$$w(n) = \{aw(n-1)\,[1+\epsilon(n)] + x(n)\}\,[1+\xi(n)] \quad . \tag{II-46}$$

The random variables $\epsilon(n)$ and $\xi(n)$ account for the roundoff errors due to the floating point multiply and add, respectively. Following Kaneko and Liu, we define the output noise sequence $e(n) = w(n) - y(n)$, subtract (II-45) from (II-46), neglect second-order terms in e, ϵ, and ξ, and obtain a difference equation for the error $e(n)$, as

$$e(n) - ae(n-1) = ay(n-1)\,[\epsilon(n) + \xi(n)] + x(n)\,\xi(n) = u(n) \quad . \tag{II-47}$$

Assuming (see Sec. I-C-2) that the $\epsilon(n)$ and $\xi(n)$ are zero mean and independent from sample-to-sample (white), and that $\epsilon(n)$, $\xi(n)$, and the signal x(n) are mutually independent, u(n) in (II-47) is white noise with variance dictated by the statistics of the signal x(n), and the variances of the roundoff variables. We assume that the variances of all roundoff variables are equal, with value denoted by σ_ϵ^2. An order-of-magnitude estimate of $\sigma_\epsilon^2 \approx 2^{-2t}/3$ can be obtained by assuming that $\epsilon(n)$ is uniformly distributed between its limits, $\pm 2^{-t}$. Empirical values of σ_ϵ^2 are slightly less than this, and the value

$$\sigma_\epsilon^2 = (0.23)2^{-2t} \tag{II-48}$$

will be used in comparing theory and experiment.

Now, if we assume that x(n) is stationary, then u(n) is stationary white noise, and by using the method of Sec. II-C-1, the variance σ_ϵ^2 of the output noise e(n) can be obtained easily from the variance σ_u^2 of u(n) as

$$\sigma_e^2 = \sigma_u^2 \sum_{n=0}^{\infty} h^2(n) = \frac{1}{1-a^2}\,\sigma_u^2 \tag{II-49}$$

where $h(n) = a^n$ is the filter impulse response.

At this point, in order to proceed any further, we must assume specific statistics for the input signal x(n). Unlike the fixed point case, we cannot obtain output noise variance without assuming signal statistics.

Let us assume, for example, that x(n) is stationary white noise of variance σ_x^2. Then, we obtain

$$\sigma_e^2 = \frac{1+a^2}{(1-a^2)^2}\,\sigma_\epsilon^2\sigma_x^2 \quad . \tag{II-50}$$

Noting that $\sigma_y^2 = \sigma_x^2/(1-a^2)$, we can express (II-50) as a noise-to-signal ratio

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{1+a^2}{1-a^2}\,\sigma_\epsilon^2 \tag{II-51a}$$

which becomes for the high-gain case, where $a = 1 - \delta$,

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{1}{\delta}\,\sigma_\epsilon^2 \quad . \tag{II-51b}$$

As might be expected, the noise-to-signal ratio increases as the pole moves toward the unit circle. The rate of increase, however, is not so great as in the case [see (II-26b)] where a fixed point filter is excited by a white signal.

As another example, x(n) was taken to be a sine wave of the form $A \sin(\omega_o n + \varphi)$ with φ uniformly distributed in $(0, 2\pi)$. The results are

$$\sigma_e^2 = \frac{A^2(1+a^2)\,\sigma_\epsilon^2}{2(1-a^2)\,(a^2 - 2a\cos\omega_o + 1)} \tag{II-52}$$

and

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{1+a^2}{1-a^2}\,\sigma_\epsilon^2 \quad . \qquad \text{(II-53)}$$

The noise-to-signal ratio turns out to be the same as for the white noise input.

To test the model, σ_e^2 was measured experimentally for white noise and sine wave inputs. The sine wave case was included because it was felt that our assumption of uncorrelated roundoff variables was most in doubt for the case of a highly correlated signal. To carry out the measurements, each input was applied to a filter using a 27-bit mantissa, and also to a filter with the same coefficient a, but using a shorter (e.g., 12-bit) mantissa in the computation. The outputs of the two filters were then subtracted, squared, and averaged over a sufficiently long period to obtain a stable estimate of σ_e^2. Because the measurement time needed for a stable estimate increases sharply as the pole of the filter approaches $z = 1$, 0.95 was the largest value used for a.

In order to compare theory and experiment, it was necessary to use a numerical value for σ_ϵ^2. Direct measurements on roundoff error sequences showed that σ_ϵ^2 for a multiplication was slightly different than for an addition, and that σ_ϵ^2 varied somewhat depending on the signal used and (for multiplication) on the coefficient a. However, it was found that rms output noise could be predicted within less than one bit, if these slight differences were neglected and the empirical "average" σ_ϵ^2 of (II-48) was used.

In Fig. 5, experimental curves for noise-to-signal ratio are compared with the predictions of the model. The same theoretical curve suffices for both white and sinusoidal inputs, since

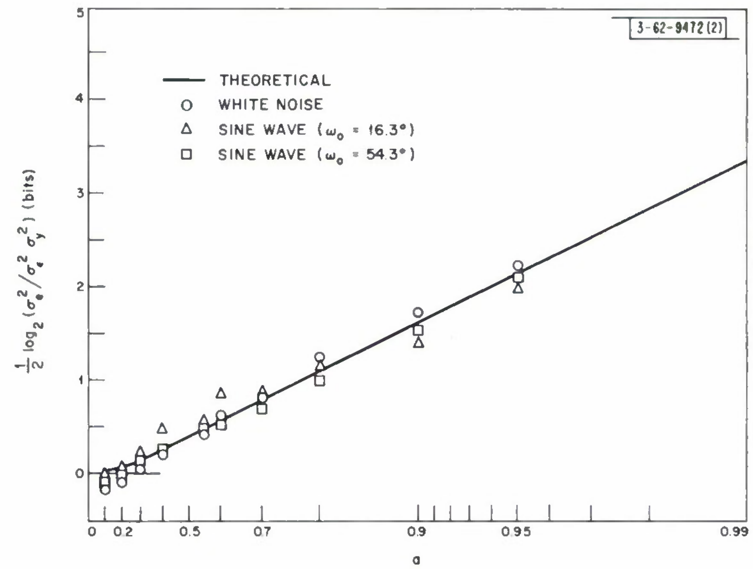


Fig. 5. Theoretical and experimental noise-to-signal ratio for floating point, first-order filter, as a function of pole position. Noise-to-signal ratio represented in bits.

(II-51a) and (II-53) are identical. The quantities actually displayed on the curve are values of

$$\frac{1}{2}\log_2\left(\frac{\sigma_e^2}{\sigma_\epsilon^2\sigma_y^2}\right)$$

a normalized rms noise-to-signal ratio represented in units of bits. We note that the agreement of theory with experiment is generally quite good.

2. Second-Order Case

An analysis similar to the above can be carried out for a second-order filter of the form

$$y(n) = r^2\, y(n-2) + 2r\cos\Theta\, y(n-1) + x(n) \tag{II-54}$$

with a complex conjugate pole pair at $z = r\,e^{\pm j\Theta}$.

When x(n) is stationary white noise, we obtain for the variance of the noise e(n),

$$\sigma_e^2 = \sigma_\epsilon^2\sigma_x^2\left[G_1 + G_1^2\left(3r^4 + 12r^2\cos^2\Theta - 16\,\frac{r^4\cos^2\Theta}{1+r^2}\right)\right] \tag{II-55}$$

where

$$G_1 = \left(\frac{1+r^2}{1-r^2}\right)\left(\frac{1}{r^4 + 1 - 4r^2\cos^2\Theta + 2r^2}\right)$$

as in (II-29b). Noting that $\sigma_y^2 = G_1\sigma_x^2$, we can express the noise-to-signal ratio as

$$\frac{\sigma_e^2}{\sigma_y^2} = \sigma_\epsilon^2\left[1 + G_1\left(3r^4 + 12r^2\cos^2\Theta - 16\,\frac{r^4\cos^2\Theta}{1+r^2}\right)\right] \tag{II-56a}$$

which, for high-gain $r = 1 - \delta$, becomes

$$\frac{\sigma_e^2}{\sigma_y^2} = \sigma_\epsilon^2\left(\frac{3 + 4\cos^2\Theta}{4\delta\sin^2\Theta}\right) . \tag{II-56b}$$

These formulas also describe closely the output noise-to-signal ratio for a two-pole, one-zero system as in (II-27), realized via the canonic form (II-31), since implementation of the zero after the pole pair has negligible effect on the noise-to-signal ratio, especially for the high-gain case. The result for the direct form (II-28) is essentially the same, and if the order of operations in (II-28) is such that $x(n) - r\cos\Theta\, x(n-1)$ is computed before adding the effects of the past output samples, then with the high-gain approximation the noise-to-signal ratio for the direct form will be exactly the same as (II-56b).

As in the fixed point case, a coupled form realization (II-42) of the two-pole, one-zero network yields an improved noise-to-signal ratio at low resonant frequencies. Assuming white input and the high-gain approximation, the result for the coupled form (the lengthy derivation is omitted) is

$$\frac{\sigma_e^2}{\sigma_y^2} = \sigma_\epsilon^2\left(\frac{3 + 4\sin^2\Theta}{4\delta}\right) . \tag{II-57}$$

The factor $1/\sin^2\Theta$ of (II-56b) is not present in (II-57). We should re-emphasize, however, that implementation of the coupled form requires more computation than the direct or canonic forms.

Let us return attention to the second-order system of (II-54). In addition to the analysis for white noise input, the output noise variance has been derived for a sinusoid $A \sin(\omega_o n + \varphi)$. The result is

$$\sigma_e^2 = A^2 G_1 \sigma_\epsilon^2 \left[\frac{3}{2} r^4 |H|^2 + 6r^2 \cos^2\Theta |H|^2 + \frac{1}{2} - 4r^3 |H|^2 \cos\Theta \cos\omega_o - r^2 |H| \cos(\Psi - 2\omega_o) + 2r |H| \cos\Theta \cos(\Psi - \omega_o)\right] \tag{II-58}$$

where $|H|$ and Ψ represent the magnitude and phase of the filter system function at the input frequency ω_o.

TABLE I

THEORETICAL AND EXPERIMENTAL NOISE-TO-SIGNAL RATIO FOR FLOATING POINT, SECOND-ORDER FILTER, AS A FUNCTION OF POLE POSITIONS

		$\frac{1}{2}\log_2(\sigma_e^2/\sigma_\epsilon^2\sigma_y^2)$ (bits)			
		White Noise		Sine Wave	
r	θ	Theoretical	Experimental	Theoretical	Experimental
0.55	22.5	1.48	1.66	1.54	1.64
0.7	22.5	2.16	2.33	2.23	2.38
0.9	22.5	3.32	3.33	3.35	3.45
0.55	45.0	0.93	1.08	0.97	0.94
0.7	45.0	1.36	1.44	1.37	1.51
0.9	45.0	2.28	2.51	2.22	2.14
0.55	67.5	0.42	0.46	0.39	0.33
0.7	67.5	0.75	0.88	0.65	0.62
0.9	67.5	1.63	1.97	1.45	0.99

By using both white noise and sinusoidal inputs, experimental noise measurements were performed for the two-pole system (II-54). In Table I, a comparison of theoretical and experimental results for output noise-to-signal ratio is displayed; (II-48) was assumed. The agreement of theory with experiment is generally quite good.

E. ROUNDOFF NOISE – BLOCK FLOATING POINT

A. V. Oppenheim[26] has proposed a realization of recursive filters using block floating point arithmetic, and analyzed the effect of roundoff errors on such a realization. Here, we review some of his work to set the stage for a comparison with fixed and floating point arithmetic.

In block floating point arithmetic, the input and filter states (i. e., the inputs to the delay registers) are jointly normalized before the multiplications and additions are performed with fixed point arithmetic. The scale factor (or exponent) obtained during the normalization is then applied to the final output to obtain a fixed point output.

A block floating point realization of the first-order filter

$$y(n) = x(n) + ay(n-1) \tag{II-59}$$

can be represented as

$$y(n) = \frac{1}{A(n)} \left\{ \left[\frac{A(n)}{A(n-1)}\right] A(n-1)\, x(n) + a \left[\frac{A(n)}{A(n-1)}\right] A(n-1)\, y(n-1) \right\} \tag{II-60}$$

where A(n) represents the scale factor used to jointly normalize x(n) and y(n − 1). A first guess at a suitable A(n) might be

$$A(n) = \frac{1}{\max\left[|x(n)|, |y(n-1)|\right]} \quad . \tag{II-61}$$

In practice, A(n) would probably be restricted to be a power of two so that a left shift rather than a multiplication could be used in the normalization. Also, depending on the order of the filter, a somewhat smaller value of A(n) than in (II-61) would be needed to prevent overflow; for a first-order filter, halving the value of A(n) given in (II-61) would prevent overflow. Figure 6 is a circuit diagram representing (II-60). The operation is as follows. The input sample x(n) is multiplied by the previous scale factor A(n − 1), then this quantity A(n − 1) x(n) is jointly normalized with the contents of the delay, A(n − 1), to determine the incremental scale factor Δ(n) = A(n)/A(n − 1). A(n) y(n) is then computed, stored in the delay, and multiplied by 1/A(n) to yield a fixed point output sample. We are then ready for a new input sample.

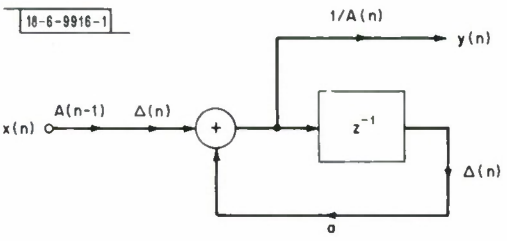


Fig. 6. Block diagram representing block floating point, first-order filter.

A roundoff noise source is introduced in the multiplication of A(n − 1) y(n − 1) by Δ(n), the multiplication by a, and the multiplication by 1/A(n). Assuming that all noise sources have zero mean and equal variance σ_ξ^2 and are independent of A(n), the output noise is

$$\sigma_e^2 = \sigma_\xi^2 \left\{ 1 + \frac{1+a^2}{1-a^2} \mathcal{E}\left[\frac{1}{A(n)}\right]^2 \right\} \tag{II-62}$$

where $\mathcal{E}$ denotes expected value. Later, for comparison with fixed and floating point, we shall assume $\sigma_\xi^2 = (1/12)2^{-2t}$, as in fixed point.

To compare this result with fixed and floating point, we need a formula for output noise-to-signal ratio. The block floating point filter yields a fixed point output, and this output must fit within a register. There, the constraint on input signal is as described in (II-22), and it is reasonable to assume a white input with variance as given in (II-24). The result for output noise-to-signal ratio is

$$\frac{\sigma_e^2}{\sigma_y^2} = \sigma_\xi^2 \left\{ \frac{3(1-a^2)}{(1-a)^2} + \frac{3(1+a^2)}{(1-a)^2} \mathcal{E}\left[\frac{1}{A(n)}\right]^2 \right\} \quad . \tag{II-63a}$$

If we assume the high-gain case where, with

$$A(n) = \frac{1/2}{\max\,[\,|x(n)|\,,\,|y(n-1)|\,]} \approx \frac{1}{2|y(n)|}$$

so that $\mathcal{E}[1/A(n)]^2 = 4\sigma_y^2$, then, with $a = 1 - \delta$, the result is

$$\frac{\sigma_e^2}{\sigma_y^2} \approx \frac{10}{\delta}\,\sigma_\xi^2 \quad . \tag{II-63b}$$

Similarly, a block floating point realization can be defined and a noise analysis carried out for the second-order filter

$$y(n) = x(n) + 2r\cos\Theta\; y(n-1) - r^2 y(n-2) \quad .$$

The output noise-to-signal ratio (for white input) is

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{3\sigma_\xi^2}{G_1}\left[\frac{1}{\sin\Theta}\sum_{n=0}^{\infty} r^n\,|\sin(n+1)\,\Theta|\right]^2$$

$$+\ \frac{G_1\sigma_\xi^2}{\sigma_y^2}\ \mathcal{E}\left[\frac{1}{A(n)}\right]^2\,(2 + 2r^4 + 4r^2\cos^2\Theta) \tag{II-64}$$

which for the high-gain case and small Θ can be approximately bracketed according to

$$\frac{1}{\delta}\left[3 + \frac{8}{\sin^2\Theta}\right]\sigma_\xi^2 \leqslant \frac{\sigma_e^2}{\sigma_y^2} \leqslant \left[12 + \frac{8}{\sin^2\Theta}\right]\sigma_\xi^2 \quad . \tag{II-65}$$

The results of noise measurements performed by Oppenheim on block floating point filters have corresponded well with the formulas just presented.

F. FIXED VS FLOATING VS BLOCK FLOATING POINT

Having obtained experimentally verifiable noise-to-signal ratios for fixed, floating, and block floating point recursive filters, we are now in a position to make a comparison. We should emphasize that our comparison is on a very restricted basis. First, many factors other than noise-to-signal ratio — such as computing facilities available, hardware requirements, ease of programming — enter into a choice of type of arithmetic. Second, we shall be comparing noise-to-signal ratios only for white input, and the results for other types of signals will, in general, be different. For example, σ_e^2/σ_y^2 for a high-gain, first-order, fixed point filter, with low-frequency (rather than white) input will tend to be proportional to $1/\delta$, rather than to $1/\delta^2$ [compare (II-26b) and (II-26d)]. If we assume low-frequency input for the floating point case, the $1/\delta$ dependence in (II-51b) will still hold. Thus, at least for this example, the fixed point case compares more favorably to the floating point case if an in-band, rather than white, signal is assumed. We proceed with our white signal based comparison, with the reservation that one must be careful in deciding whether the comparison is valid for his particular application.

For a first-order filter, and white input, let us summarize the results derived above for output noise-to-signal ratio:

$$\left.\frac{\sigma_e^2}{\sigma_y^2}\right)_{\text{fixed point}} = \frac{1}{4(1-a)^2}\,2^{-2t} \tag{II-66}$$

$$\left.\frac{\sigma_e^2}{\sigma_y^2}\right)_{\text{floating point}} = \frac{(0.23)\,(1+a^2)}{(1-a^2)}\,2^{-2t} \tag{II-67}$$

$$\left.\frac{\sigma_e^2}{\sigma_y^2}\right)_{\text{block floating}} = \left[\frac{(1+a)}{4(1-a)} + \frac{(1+a^2)}{4(1-a)^2}\,k\right]2^{-2t} \tag{II-68}$$

where

$$k = \mathcal{E}\{[1/A(n)]^2\} \tag{II-69a}$$

and

$$A(n) = \frac{1/2}{\max\,[|x(n)|,\,|y(n-1)|]} \quad . \tag{II-69b}$$

The quantity k in (II-68) is signal dependent. For high gain, we would expect $k \approx 4\sigma_y^2$. Oppenheim has measured experimentally the ratio

$$k' = k/4\sigma_y^2 \tag{II-70}$$

for block floating point filters with the coefficient ranging from $a = 0.1$ to 0.95. The values of k' were all between 2 and 3. Rewriting (II-68) in terms of the quantity k', we have

$$\left.\frac{\sigma_e^2}{\sigma_y^2}\right)_{\text{block floating}} = \left[\frac{(1+a)}{4(1-a)} + \frac{(1+a^2)}{3(1-a^2)}\,k'\right]2^{-2t} \quad . \tag{II-71}$$

Figure 7 compares curves of output noise-to-signal ratio as given by (II-66), (II-67), and (II-71). Actually, we plot in each case the quantity $\frac{1}{2}\log_2(\sigma_e^2/2^{-2t}\sigma_y^2)$, representing an rms output noise-to-signal ratio, in units of bits. For the block floating point case, Oppenheim's experimental values of k' were used up to $a = 0.95$; for $a \geqslant 0.99$, the high-gain approximation $k' = 1$ was used. From Fig. 7, we observe that floating point arithmetic leads to the smallest noise-to-signal ratio, block floating point the next smallest, and fixed point the largest. We also notice that for high-gain filters, as a increases toward unity, the noise-to-signal ratio for fixed point increases faster than for floating or block floating point.

This comparison, however, is based on the assumption that the number of bits in the mantissa is the same for the three types of arithmetic. The comparison does not account for the additional bits needed for the characteristic in either block floating or floating point arithmetic.

In floating point, all quantities are represented with a mantissa and characteristic. If c denotes the number of bits in the characteristic, this could be accounted for in Fig. 7 by numerically adding the constant c to the floating point data. This shift will cause the floating and fixed point curves to cross at a point where the noise-to-signal ratios are equal for equal total register lengths. To compare fixed and floating point on this basis, we provide just enough bits in the characteristic to allow the same dynamic range for both the fixed and floating point filters. If t_{fx} denotes the length of the fixed point fraction, then the requirement of identical dynamic range implies that

$$c = \log_2 t_{fx} \quad . \tag{II-72}$$

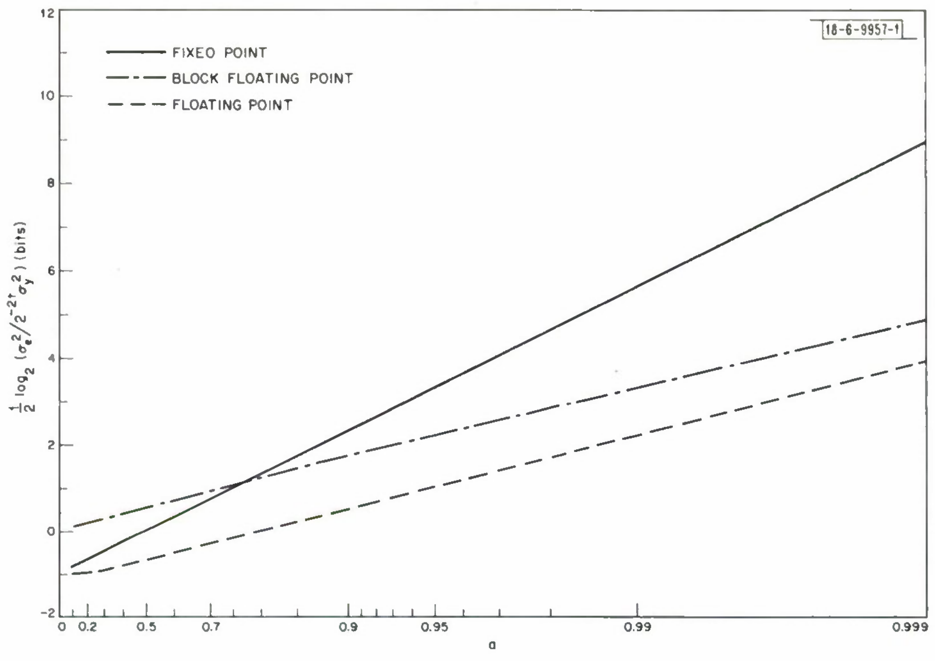


Fig. 7. Comparison of fixed, floating, and block floating point noise-to-signal ratios — first-order filter.

By assuming, for example, that $t_{fx} = 16$ so that $c = 4$, a crossover point in noise-to-signal ratio will occur at $a = 0.996$.

For block floating point, one exponent is used for both $x(n)$ and $y(n-1)$, so it is not as straightforward to compare fixed and block floating point on the basis of equal register lengths. However, the issues in such a comparison should be clear.

Now let us consider the second-order case, specifically a filter of the form (II-54) with a complex conjugate pole pair. For white input, our results for output noise-to-signal ratio can be summarized as:

$$\left.\frac{\sigma_e^2}{\sigma_y^2}\right)_{\text{fixed point}} = \frac{1}{2}\left\{\frac{1}{\sin\Theta}\sum_{n=0}^{\infty} r^n\left|\sin[(n+1)\Theta]\right|\right\}^2 2^{-2t} \qquad \text{(II-73)}$$

$$\left.\frac{\sigma_e^2}{\sigma_y^2}\right)_{\text{floating point}} = (0.23)\left[1 + G_1\left(3r^4 + 12r^2\cos^2\Theta - 16\,\frac{r^4\cos^2\Theta}{1+r^2}\right)\right] 2^{-2t} \qquad \text{(II-74)}$$

$$\left.\frac{\sigma_e^2}{\sigma_y^2}\right)_{\text{block floating}} = \frac{2^{-2t}}{4G_1}\left\{\frac{1}{\sin\Theta}\sum_{n=0}^{\infty} r^n\left|\sin[(n+1)\Theta]\right|\right\}^2 + \frac{2^{-2t}G_1}{4\sigma_y^2}$$
$$\times\, k(2 + 2r^4 + 4r^2\cos^2\Theta) \qquad \text{(II-75)}$$

where now

$$k = \mathcal{E}\{[1/A(n)]^2\} \qquad \text{(II-76a)}$$

$$A(n) = \frac{1/2}{\max\left[|x(n)|,\ |y(n-1)|,\ |y(n-2)|\right]} \quad . \qquad \text{(II-76b)}$$

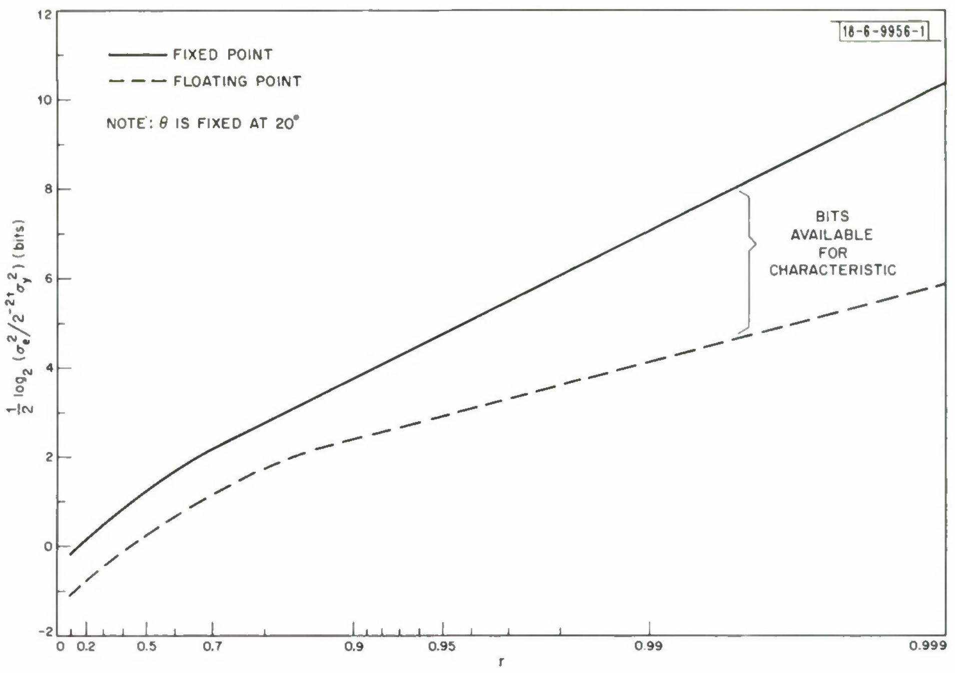


Fig. 8. Comparison of fixed and floating point noise-to-signal ratios – second-order filter.

We do not have available measured values of k for the second-order case, so in Fig. 8 we display the results only for fixed and floating point.† θ has been fixed at 20°, and r is varied. The comparison between fixed and floating point is similar to the first-order case depicted in Fig. 7. For $t_{fx} = 16$, a crossover point in noise-to-signal ratio would occur at r = 0.998, θ = 20°.

Let us re-emphasize that the fixed vs floating point comparisons of Figs. 7 and 8 are valid only for white input. As we illustrated at the end of Sec. II-C-2, $\left(\sigma_e^2/\sigma_y^2\right)_{\text{fixed point}}$ is quite dependent on the signal spectrum so that assuming different signal statistics, we might obtain a very different comparison. In Sec. II-D-1, we found that $\left(\sigma_e^2/\sigma_y^2\right)_{\text{floating point}}$ was the same for white and sinusoidal inputs, indicating that the floating point noise-to-signal ratio is relatively insensitive to changes in the signal spectrum.

† The relationship between the fixed and floating point curves in Fig. 8 is different than in Fig. 2b of Weinstein and Oppenheim,[30] because an error was made in the earlier paper, where all points on the fixed point curve should be 1.57 bits higher than the values displayed. With this correction, the earlier figure will be seen to correspond to our Fig. 8, if we note that the quantity plotted in the earlier paper was $1/2\ \log_2(\sigma_e^2/\sigma_\epsilon^2\sigma_y^2) = 1/2\ \log_2[\sigma_e^2/(0.23)2^{-2t}\sigma_y^2]$ instead of $1/2\ \log_2(\sigma_e^2/2^{-2t}\sigma_y^2)$ as in Fig. 8.

III. THE FAST FOURIER TRANSFORM (FFT)

In this section, we study the effects of quantization on implementation of the FFT algorithm. Errors due to coefficient quantization and to rounding in the computation will each be considered. Schemes for scaling the array so that no overflows occur will be discussed. The differing quantization effects in fixed, floating, and block floating point arithmetic will be examined and compared.

There are many forms of the FFT algorithm, and the detailed effects of quantization will differ depending on the form used. Our specific results will be obtained for the case of a radix 2 algorithm. Much of the theory will be valid for any of the standard radix 2 algorithms, but all experiments have been carried out using a decimation in time algorithm.[2,6] We feel that most of the ideas employed in the error analysis of simple radix 2 algorithms can be utilized in error analysis of other forms of the algorithm (mixed radix, etc.).

Our approach in analyzing noise in the FFT is basically statistical. In most cases, we support the predictions of our models with experimental data. For floating and block floating point arithmetic, in order to simplify analysis and obtain concrete results, we find it convenient to assume a simple, white model for the signal being transformed. Discussion of how the results might be expected to change for other types of signals is included, as are experimental noise measurements on FFT's of nonwhite signals.

A. COEFFICIENT QUANTIZATION

In computing a discrete Fourier transform (DFT) via the FFT, the system coefficients used are values of

$$W^k = e^{(-j2\pi/N)k} = \cos\frac{2\pi k}{N} - j\sin\frac{2\pi k}{N} \qquad \text{(III-1)}$$

for $k = 0, 1, \ldots, N-1$, where N is the number of points transformed. This implies a requirement of $2N$ real coefficients, but symmetries in the trigonometric functions can be used to reduce the number needed. These coefficients may be obtained in various ways, for example via table look up, or from the iteration of a difference equation with poles on the unit circle. But in whatever manner the coefficients are obtained, they can, because of the finite word length, only approximate the true values of the functions.

Our aim here is to estimate the error in a computed DFT, due to the errors in these system coefficients. To separate the errors due to coefficient quantization and roundoff noise, we shall assume in this discussion that all arithmetic is performed with perfect precision, though the coefficients are quantized. For the purpose of analysis, we shall assume that all coefficients in (III-1) are obtained from a stored table which is accurate to the full computer word length except for a roundoff in the last bit. Our initial discussion will pertain to fixed point arithmetic, but we shall show how, with a slight modification, the analysis can be applied also to the floating point case. We shall deal with the case of a radix 2 FFT algorithm, where $N = 2^{\nu}$, and the DFT is computed in ν stages.

To set up the problem, we first write the definition of the DFT of an N-point sequence,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W^{nk} \qquad k = 0, 1, \ldots, N-1 \quad . \qquad \text{(III-2)}$$

This can be viewed as a multiplication of the N-vector x(n) by the $N \times N$ matrix $[W^{nk}]$. Instead of (III-2), an FFT computation using rounded coefficients would yield

$$\hat{X}(k) = \sum_{n=0}^{N-1} x(n)\, \Omega_{nk} \tag{III-3}$$

where the matrix $[\Omega_{nk}]$ is different from $[W^{nk}]$. For a radix 2 algorithm, each element Ω_{nk} will be a product of ν factors obtained from our sine-cosine table. Thus,

$$\Omega_{nk} = \prod_{i=1}^{\nu} (W^{a_i} + \delta_i) \tag{III-4a}$$

where

$$\prod_{i=1}^{\nu} W^{a_i} = W^{nk} \tag{III-4b}$$

and δ_i represents a rounding error in one of our tabled coefficients. It is to be understood in (III-4) that the ν values of W^{a_i} and δ_i are, in general, different for each Ω_{nk}. The complex rounding error δ_i satisfies the bound

$$|\delta_i| \leqslant \sqrt{2}\, 2^{-t} \tag{III-5}$$

and, if assumed to obey our usual statistical assumption on roundoff errors, would have zero mean and variance

$$\mathcal{E}|\delta_i|^2 = (2)2^{-2t}/12 = \frac{2^{-2t}}{6} \quad . \tag{III-6}$$

The error E(k) in a computed DFT value can be expressed as

$$E(k) = \hat{X}(k) - X(k) = \sum_{n=0}^{N-1} x(n)\,(\Omega_{nk} - W^{nk}) = \sum_{n=0}^{N-1} x(n)\, \delta\Omega_{nk} \quad . \tag{III-7}$$

so that the error E(k) is the result of multiplication of the input sequence x(n) by the "error matrix" $[\delta\Omega_{nk}]$.

Using (III-4), (III-5), and (III-7), we can derive a deterministic bound on the elements of the error matrix, and from this obtain a deterministic bound on the ratio of mean-squared output error to mean-squared output signal. However, the bound obtained is proportional to $N\nu^2 2^{-2t}$, which as we shall see below is quite pessimistic compared with results which normally occur.

Using instead a random model for the error matrix, we can estimate statistically the expected output noise-to-signal ratio. The predictions of the model can then be checked against experimental error measurements. This approach is now discussed.

From (III-4) and (III-7), we can obtain an expression for an element $\delta\Omega_{nk}$ of the error matrix

$$\delta\Omega_{nk} = \sum_{i=1}^{\nu} \delta_i \prod_{\substack{j=1 \\ j\neq i}}^{\nu} W^{a_i} + \text{higher order terms} \tag{III-8}$$

where the higher order terms involve products of rounding errors, and are thus of the order 2^{-2t}. Assuming t sufficiently large so that these higher order terms can be neglected, observing that $|W| = 1$, assuming the δ_i mutually uncorrelated, and using (III-6), we obtain

$$\mathcal{E}|\delta\Omega_{nk}|^2 = \upsilon \frac{2^{-2t}}{6} \quad . \tag{III-9}$$

Making the further assumption that all elements Ω_{nk} are uncorrelated with each other and with the signal x(n), we obtain

$$\mathcal{E}|E(k)|^2 = \sigma_E^2 = \upsilon \frac{2^{-2t}}{6} \sum_{n=0}^{N-1} |x(n)|^2 \tag{III-10}$$

or by Parseval's relation

$$\frac{\sigma_E^2}{\frac{1}{N} \sum_{n=0}^{N-1} |X(k)|^2} = \upsilon \frac{2^{-2t}}{6} \quad . \tag{III-11}$$

This is the desired result for the ratio of mean-squared output error to mean-squared output signal.

Some of the assumptions used in obtaining (III-11) are clearly in doubt. For example, each of the coefficients stored in our table is used a number of times in the FFT computation, and, of course, the roundoff error in this coefficient is the same each time it is used. Thus, the assumptions of mutually uncorrelated δ_i and mutually uncorrelated $\delta\Omega_{nk}$ are quite questionable. Further, we have assumed that all transmissions in our FFT flow chart are along paths where multiplication by an inaccurate coefficient is required, whereas in many cases only an addition or subtraction, and thus no multiplication by an inaccurate coefficient, is required. For instance, in a standard decimation in time algorithm, the matrix element Ω_{oo} would be identically equal to 1, with no error. This might lead us to believe that our estimate (III-11) for error-to-signal ratio might be slightly high.

Although we might not expect (III-11) to predict with great accuracy the error in an FFT due to coefficient quantization, it would be helpful even as a rough estimate of the error. The key result of (III-11), which we would like to test experimentally, is that the error-to-signal ratio increases very mildly with N, being proportional to $\upsilon = \log_2 N$, so that doubling N produces a very slight increase in error-to-signal ratio. When we consider later the output noise due to roundoff, we shall find (for the fixed point case) a much sharper increase with N of the output noise-to-signal ratio.

To test this result, experimental measurements on errors due to coefficient quantization were made. In each run, a sequence x(n) — white in the sense that all 2N real numbers making up the N-point complex sequence were mutually uncorrelated, with zero means, and equal variances — was obtained using a random number generator. This sequence was transformed twice, once using a 36-bit coefficient table, and once using a coefficient table rounded to much shorter word length (e.g., 12 bits). For each transform, 36-bit accuracy was used in the arithmetic to make the effect of roundoff error negligible. The results were subtracted, squared, and divided by the output signal variance (N times the input signal variance) to obtain an experimental output error-to-signal ratio. For each value of N, several random sequences were transformed and the results were averaged to obtain statistically convergent estimates.

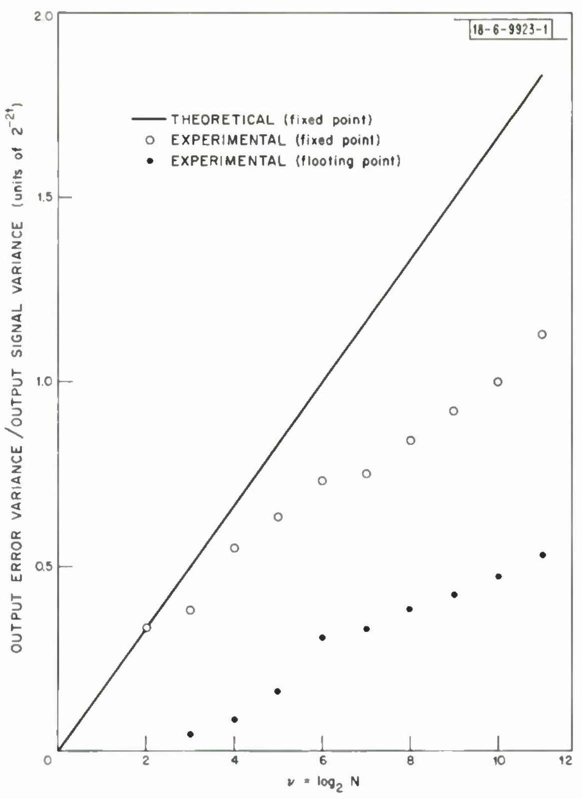


Fig. 9. Errors due to coefficient quantization in FFT.

These results are displayed in Fig. 9; the quantity plotted is $\sigma_E^2/2^{-2t}\sigma_X^2$, where $\sigma_X^2 = \mathcal{E}|X(k)|^2$. The theoretical curve

$$\frac{\sigma_E^2}{2^{-2t}\sigma_X^2} = \frac{\upsilon}{6} \tag{III-12}$$

corresponding to (III-11), is shown, and the circles represent measured output error-to-signal ratio for the fixed point case. We note that the experimental results generally lie below the theoretical curve. No experimental result differs by as much as a factor of two from the theoretical result, and since a factor of two in σ_E^2/σ_X^2 corresponds to only half-a-bit difference in the rms output error, it seems that (III-12) is a reasonably accurate estimate of the effect of coefficient errors. The experimental results do seem to increase essentially linearly with υ, but with smaller slope than given in (III-12).

In the above, fixed point arithmetic has been assumed. However, since a block floating point FFT will generally use fixed point coefficients, our results are valid for the block floating point case also. With some slight modifications, we can obtain similar results for the floating point case.

In fact, the argument leading to (III-11) applies directly to the floating point case, except that for floating point we must consider that the expected magnitude of the rounding error in a tabled coefficient is proportional to the magnitude of the coefficient. Thus, the tabled coefficient approximating W^k will have the rounded value

$$\cos\frac{2\pi k}{N}(1+\epsilon) - j\sin\left(\frac{2\pi k}{N}\right)(1+\xi) \quad .$$

If we assume random roundoff variables ϵ and ξ, with variance $\sigma_\epsilon^2 = \sigma_\xi^2$, then the roundoff error

$$\delta = \epsilon \cos \frac{2\pi k}{N} - j\xi \sin \frac{2\pi k}{N}$$

has variance

$$\mathcal{E}|\delta|^2 = \sigma_\epsilon^2 \quad . \tag{III-13}$$

With (III-13) replacing (III-6), the remainder of the argument leading to (III-11) and (III-12) follows, and we have for the floating point case

$$\frac{\sigma_E^2}{\sigma_X^2} = \sigma_\epsilon^2 \upsilon \quad . \tag{III-14}$$

Except for perhaps a constant factor, the floating and fixed point results are the same, since σ_ϵ^2 is proportional to 2^{-2t}. Experimental results for the floating point case are represented by the solid dots in Fig. 9, and are observed to be slightly lower than the results for the fixed point case.† As a partial accounting for this difference, we can mention that in the floating point program, the frequently used coefficients $W^k = 1$ and $-j$ were represented exactly, whereas this was not the case for our fixed point program. However, the discrepancies between the results are very slight from a practical viewpoint, since a factor of two difference in σ_E^2/σ_X^2 represents only a half-a-bit difference in rms output error.

The above work was designed only to give a rough estimate of the errors to be expected, on the average, due to coefficient quantization in the FFT. Many phenomena, which occur in special cases, cannot be explained by our statistical model. For example, it has been observed (for our decimation in time algorithm, with tabled coefficients) that if x(n) is a single pulse at n = 0, X(k) will be a constant, despite coefficient quantization. Similarly, a constant x(n) yields a single pulse at k = 0 for X(k). It has been found that a single frequency input $x(n) = e^{(j2\pi ln/N)}$ will yield, because of coefficient quantization, spurious peaks at multiples of the input frequency. Also, it has been observed that a real input x(n) yields a transform whose real part is even and whose imaginary part is odd, despite the coefficient quantization. Each of these phenomena can be explained by considering special symmetries and other details of the actual FFT computation. However, the explanations are each valid only for a specific input and a specific algorithm, and are not particularly enlightening from a general viewpoint.

Some comments can be made about the case where coefficients are obtained from a recursive difference equation, rather than from a table look-up. First, one would probably be wise to use a coupled form, rather than a direct form digital oscillator, since the former yields much greater accuracy in the sine and cosine values.[21] Even using the coupled form, one should frequently reset the initial conditions of the oscillator. As the oscillator is allowed to run for longer periods, the sine and cosine values obtained tend to become more in error from the desired values, both because of roundoff error in computing the recursion and (perhaps more importantly) because the coefficients of the recursion will be slightly in error, yielding an error in output frequency. This effect has been observed for a 2048-point transform, in which the digital oscillator was reset only at the beginning of each stage of the FFT. By the last stage, when the

† In comparing the results as plotted on the same graph, it is implicitly assumed that the number of bits in the floating point mantissa is equal to the number of bits in the fixed point fraction. Actually, the need for an exponent then implies greater total word length for floating point.

recursion was run for 1024 points, some errors in the generated coefficients were large enough to affect the last 10 bits of these coefficients. Significant distortions in computed transforms were observed to result from these errors. However, more frequent resetting of the initial conditions of the coefficient-producing recursions would significantly reduce this error.

B. ROUNDOFF NOISE AND OVERFLOW CONSTRAINT – FIXED POINT

In this section, we analyze the effects of roundoff errors and the overflow constraint to obtain formulas for output noise-to-signal ratio in a fixed point FFT. At first, we ignore the overflow constraint and use the general approach previously applied to recursive filters (see Sec. II-C-1) to obtain formulas for output noise variance. Then, we consider the overflow constraint and derive output noise-to-signal ratio formulas. We find that if overflow is prevented by attenuating the array at each stage of the FFT, rather than simply by one large attenuation of the initial input array, the output noise-to-signal ratio is improved. Experimental verification of the predictions of the noise model are presented. The results in this section are essentially the same as the corresponding results reported by Welch,[27] but the route taken in the analysis is somewhat different.

First, we recall our approach to analysis of fixed point roundoff noise. On a signal flow graph of the system, we insert additive, signal-independent, white noise sources after each multiplier. To determine the effect of any individual noise source, we find (via inspection of the flow graph) a transfer function between the noise source and the system output, and use linear system noise theory to derive the corresponding output noise statistics. Then, applying our assumption that all noise sources are independent of each other, we determine overall output noise statistics by simply adding the effects of all the individual noise sources. In the simple recursive filters studied earlier, only a few noise sources needed to be considered, whereas in the FFT we have many noise sources with which to deal. An added complication is that, since the computation leading to each FFT output point is different, it might seem necessary to perform a separate noise analysis for each output point. However, we shall see that the properties of this remarkable algorithm (the FFT) are such that these complications can be dealt with neatly, and simple results can be obtained.

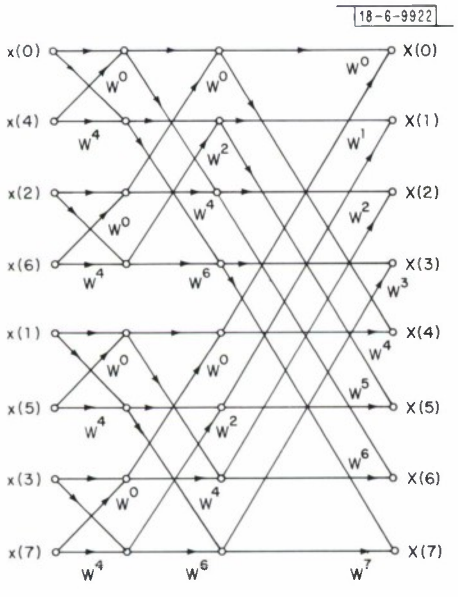


Fig. 10. FFT flow graph, N = 8.

Figure 10 displays an FFT flow graph for N = 8. A specific decimation in time algorithm is depicted. (An implementation of this particular form of the algorithm was used for all our experimental work.) Some key aspects of this diagram, which are common to all standard radix 2 algorithms, are as follows. The DFT is computed in $\nu = \log_2 N$ stages. At each stage, the algorithm passes through the entire array of N complex numbers, two at a time, generating a new N number array. The νth array contains the desired DFT. The basic numerical computation operates on a pair of numbers in the mth array, to generate a pair of numbers in the $(m + 1)$st array. This computation, referred to as a "butterfly," is defined by

$$X_{m+1}(i) = X_m(i) + \widetilde{W}X_m(j) \tag{III-15a}$$

$$X_{m+1}(j) = X_m(i) - \widetilde{W}X_m(j) \quad . \tag{III-15b}$$

Here, $X_m(i)$ and $X_m(j)$ represent a pair of numbers in the m^{th} array, and $\widetilde{W}$ is some appropriate integer power of W, that is,

$$\widetilde{W} = W^p = e^{-j2\pi p/N} \quad . \tag{III-16}$$

At each stage, N/2 separate butterfly computations such as (III-15) are carried out to produce the next array. The integer p varies with i, j, and m in a complicated way, which depends on the specific form of the FFT algorithm that is used. Fortunately, our analysis is not tied to the specific way in which p varies. Also, the specific relationship between i, j, and m, which determines how we index through the m^{th} array, is not important for our analysis.

Given an FFT flow graph, we can model the roundoff noise by associating an independent white noise generator with each multiplier. This means that a noise source feeds into each node of the signal flow graph of Fig. 10 (excluding the initial array of nodes, since we are not considering A-D noise here). Since we are dealing with complex multiplications, these elemental noise sources are complex. Defining the complex variance σ_B^2 as the expected squared magnitude of such a noise source, we have

$$\sigma_B^2 = 4\sigma_o^2 = 4\,\frac{2^{-2t}}{12} \tag{III-17}$$

where we have assumed that each of the four real multiplications used to perform the complex multiplication is rounded separately. In Fig. 10, $3 \times 8 = 24$ such noise sources must be inserted.

Our next objective is to add up the effects of all these noise sources to obtain the output noise variance. The key observation which simplifies this is that the transmission function from any node in the flow graph to any other connected node is simply a multiplication by a complex constant of unity magnitude (i.e., a power of W). Thus, the noise variance at any node in the output array, due to a noise source at any particular node in the flow chart, will be equal to the noise variance σ_B^2 if the two nodes are connected, or equal to zero if the noise source node is not connected to this particular output node. Since we are assuming all noise sources uncorrelated, we can compute the noise variance at any output node by simply counting the number of noise sources that connect to this node, and multiplying this number by σ_B^2.

This count can be made by inspection of the flow chart, as illustrated in Fig. 11. The output point X(2) is singled out, and the heavy solid lines define a "tree" whose branches double in number as we proceed backward from array to array in the flow chart. At the nodes of

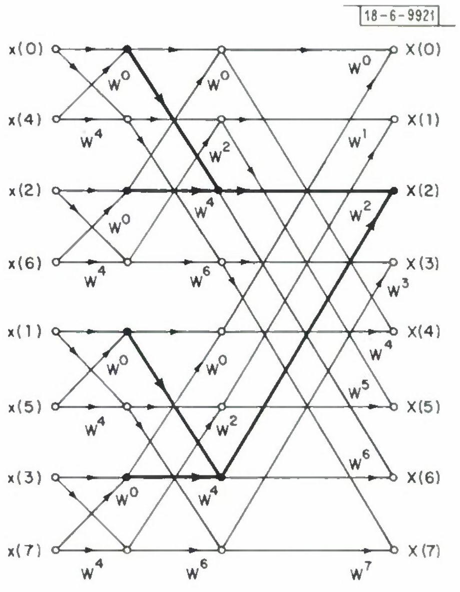


Fig. 11. FFT flow graph, N = 8. Solid lines (tree) indicate those noise sources which affect a particular output point.

this tree are the noise sources that connect to this particular output point. The tree has

$$1 + 2 + 4 = 7 \text{ nodes}$$

so the variance of the output noise E(2), at this particular output point, is

$$\mathcal{E}|E(2)|^2 = \sigma_E^2 = 7\sigma_B^2 \quad .$$

We can easily verify that a similar tree drawn backward from any output point will have the same number of nodes, so that

$$\sigma_E^2 = (1 + 2 + 4)\,\sigma_B^2 = 7\sigma_B^2$$

is the noise variance for any output point. This argument can be easily extended to FFT's for larger numbers of points, since our tree simply doubles its number of branches for each additional stage. Thus, for $N = 2^\nu$, we have

$$\sigma_E^2 = (1 + 2 + 4 + \ldots + N/2)\,\sigma_B^2$$

or

$$\sigma_E^2 = (N - 1)\,\sigma_B^2$$

which, for reasonably large N, we shall take as

$$\sigma_E^2 = N\sigma_B^2 \quad . \qquad \text{(III-18)}$$

This is the basic result we have been seeking for output noise variance in the FFT. It says that the variance of the output noise is independent of position in the output array, and is proportional to N, the number of points transformed. The effect of doubling N, or adding another stage in the FFT, is to double the output noise variance. Using the assumptions we have made thus far about the noise generators in the FFT (all uncorrelated, with equal variances), we can deduce a further result — namely, that the output noise is white, i.e., the N noise samples E(k) are mutually uncorrelated, with independent real and imaginary parts. One can prove this easily by first being convinced that the output of any butterfly is white (two outputs uncorrelated with equal variance, real and imaginary parts uncorrelated) if the input is white. Since the noise sources in our system are white, and all connected to the output via some combination of butterfly computations, the output noise must also be white.

In order to simplify the analysis leading to (III-18), we have neglected some details. First, we have associated equal variance noise sources with all multipliers, including where $\widetilde{W} = 1$ and j. In many programmed FFT's these multiplications are performed noiselessly. If we assume in the analysis that these multiplications are noiseless, the output noise variance will no longer be uniform over the output array. (For example, the zeroth output point would be noiseless.) The average variance over the output array will be somewhat lower than the result in (III-18), but will retain a linear dependence on N. Second, the assumption that all noise sources are uncorrelated is contradicted by the fact that the two noise sources associated with a given butterfly are negatives of each other, and therefore completely correlated. This does not affect the result for output noise variance, since the two outputs of a butterfly connect to a disjoint set of output points. However, it implies that the output noise samples E(k) are somewhat correlated.

We feel that these details are worth mentioning, but not worth analyzing here at length, because they cloud the essential ideas of the analysis, are quite program-dependent, and do not change the essential character of the dependence of mean-squared output noise on N.

We now wish to obtain a formula for output noise-to-signal ratio in an FFT, by considering the overflow constraint in conjunction with our noise analysis. We can insure against overflow in the FFT computation by keeping the input x(n) sufficiently small so that no output X(k) exceeds unity in magnitude. The maximum modulus of the complex numbers in the array is nondecreasing as we proceed from stage to stage, since we can deduce easily from (III-15) that

$$\max \{|X_m(i)|, |X_m(j)|\} \leqslant \max \{|X_{m+1}(i)|, |X_{m+1}(j)|\}$$
$$\leqslant 2 \max \{|X_m(i)|, |X_m(j)|\} \quad . \qquad \text{(III-19)}$$

Thus, the constraint $|X(k)| < 1$ guarantees also that there will be no overflow in the computation of an intermediate array.

In order to express this constraint as a bound on the input sequence, we observe (from the definition of the DFT) that the maximum possible output can be expressed in terms of the maximum input as

$$|X(k)|_{\max} = |x(n)|_{\max} \sum_{n=0}^{N-1} |W^{nk}| = N|x(n)|_{\max} \quad . \qquad \text{(III-20)}$$

Thus, bounding the input sequence so that

$$|x(n)| < 1/N \qquad \text{(III-21)}$$

will prevent overflow. To obtain an explicit expression for output signal variance, we assume x(n) white, with real and imaginary parts each uniformly distributed in $(-1/\sqrt{2}\ N,\ 1/\sqrt{2}\ N)$. Then, we have

$$\sigma_X^2 \equiv \mathcal{E}|X(k)|^2 = N\sigma_x^2 \equiv N\,\mathcal{E}|x(n)|^2 = \frac{1}{3N} \quad . \qquad \text{(III-22)}$$

By combining this with (III-18),

$$\frac{\sigma_E^2}{\sigma_X^2} = 3N^2\sigma_B^2 \quad . \qquad \text{(III-23)}$$

The assumption of white input signal is not critical here. For example, if a complex sinusoid $x(n) = (1/N) \exp [j(2\pi k_o/N + \varphi)]$ had been selected, σ_E^2/σ_X^2 would still be proportional to N^2, which is the essential point of (III-23).

We can be slightly more clever about the way we prevent overflow in the FFT, and obtain a noise-to-signal ratio proportional to N, instead of to N^2. We observe [see (III-19)] that the maximum modulus of the numbers in an array increases gradually, never by more than a factor of two, as we proceed from stage to stage. Thus, we can prevent overflow by incorporating an attenuation factor of 1/2 at each stage, and requiring for the input array that $|x(n)| < 1$. The attenuation may be carried out previous to the computation of each succeeding stage, or may be incorporated into the butterfly computation. This step-by-step scaling tends to keep the signal level as high as possible (compared with the roundoff errors) as we proceed from stage to stage, and thus might be expected to yield lower noise-to-signal ratio than when we simply scale the input array so that $|x(n)| < 1/N$. This is indeed the case. Using step-by-step scaling, the attainable output signal level (for white input) is the same as in (III-22), since the output signal level does not depend on where the scaling is done, but only on how much overall scaling is done. However, with step-by-step scaling, the output noise level will be less than in (III-18), since the noise introduced at early stages of the FFT will be attenuated by the scaling which takes place at the later array. To consider this quantitatively, let us refer back to Fig. 11. The tree still enables us to determine those noise sources which affect a given output point, but now attenuation factors of 1/2 must be included on all the arrows of the flow graph in order to determine the output noise variance resulting from any noise source. For example, a noise source in the second-from-last array will go through an attenuation of 1/4 in propagating to the output, resulting in an output noise variance which is $(1/4)^2 = 1/16$ times the variance of the noise source. For the $N = 8$ case depicted, the output noise variance is

$$\sigma_E^2 = [1 + 2(\tfrac{1}{2})^2 + 4(\tfrac{1}{4})^2]\, \sigma_{B'}^2,$$

where $\sigma_{B'}^2$ represents the noise source variance. We notice that although earlier arrays contribute a larger number of noise sources to a given output point, their actual contribution to σ_E^2 is less than that of later arrays, because of the scale factors.

For the general case $N = 2^\nu$, we can deduce that

$$\sigma_E^2 = [1 + 2(\tfrac{1}{2})^2 + 4(\tfrac{1}{4})^2 + \ldots + \tfrac{N}{2}(\tfrac{2}{N})^2]\, \sigma_{B'}^2,$$

or approximately (for large N)

$$\sigma_E^2 = 2\sigma_{B'}^2 \quad . \tag{III-24}$$

Thus, for reasonably large N, the output noise variance does not increase with N, and is much less than the variance in (III-18). Generally, $\sigma_{B'}^2$ will be slightly higher than σ_B^2, since it must include the noise due to the scaling as well as the noise due to the multiplication by $\widetilde{W}$.

We assume that the scaling is accomplished by rounded right shift where, if the bit to be shifted off is 1, a random choice is made as to whether to round the result up or down. The rounding error is 0 with probability 1/2, $-(1/2)2^{-t}$ with probability 1/4, and $(1/2)2^{-t}$ with probability 1/4. Its variance is

$$\sigma_s^2 = \frac{2^{-2t}}{8} \quad . \tag{III-25}$$

Since shifts of both real and imaginary parts are needed, we have

$$\sigma_{B'}^2 = 4\,\frac{2^{-2t}}{12} + 4\,\frac{2^{-2t}}{8} = (5/6)2^{-2t} \tag{III-26}$$

slightly greater than $\sigma_B^2 = (4/12)2^{-2t}$.

Now, we can combine (III-24) with (III-22) to obtain the output noise-to-signal ratio for the case of step-by-step scaling. We obtain

$$\frac{\sigma_E^2}{\sigma_X^2} = 6N\sigma_{B'}^2 = (5N)2^{-2t} \tag{III-27}$$

a result proportional to N, rather than to N^2. An interpretation of (III-27) is that the rms output noise-to-signal ratio increases as $\sqrt{N}$, or by half a bit per stage. This result has been stated by Welch.[27] It is important to note that the assumption of white signal is not essential in the analysis. The basic result of half-a-bit-per-stage increase holds for a broad class of signals, with only the constant multiplier in (III-27) being signal-dependent. In general, with scaling at each array, the output variance is related to the variance of the input array by

$$\sigma_X^2 = \frac{1}{N}\sigma_x^2 = \frac{1}{N}\,\mathcal{E}|x(n)|^2 \tag{III-28}$$

so that

$$\frac{\sigma_E^2}{\sigma_X^2} = \frac{(5/3)N2^{-2t}}{\sigma_x^2} \tag{III-29}$$

where, to reduce noise-to-signal ratio, we would like to make σ_x^2 as large as possible but are limited by the constraint $|x(n)| < 1$.

We should also note that the dominant factor causing the increase of σ_E^2/σ_X^2 with N is the decrease in signal level (required by the overflow constraint) as we pass from stage to stage. According to (III-24) and (III-26), very little noise (only a bit or two) is present in the final array. Most of the noise has been shifted off by the scalings. However, the mean-squared signal level has decreased by a factor of 1/N from its initial value, due to the scalings. Our output consists not of the DFT defined by (III-2), but of 1/N times this DFT.

We have assumed straight fixed point computation in this section, i.e., only preset attenuations were allowed, and we were not permitted to rescale on the basis of an overflow test. Clearly, if our hardware or programming facility are such that straight fixed point must be used, we should if possible incorporate attenuators of 1/2 at each array, rather than using a large attenuation of the input array.

In Sec. C below, we shall consider the effect of roundoff noise on a block floating point FFT, where the computations are continually checked for overflow, and a scaling of the array is carried out only when an overflow occurs. For some inputs, scaling will not be necessary at all stages of the FFT, so that the signal level and signal-to-noise ratio can be kept higher than in fixed point. Our fixed point analysis just presented can be considered as a worst-case analysis for block floating point.

The predicted output noise-to-signal ratio, as given in (III-29), has been checked experimentally. Test sequences were transformed twice – once using 36-bit arithmetic, and once using rounded arithmetic with much shorter word length (e.g., 12 bits). The results were subtracted, squared, and averaged to obtain an estimate of noise variance. Both white and sinusoidal inputs were used as test signals. Figure 12 displays the results, plotting on a log-log scale normalized rms output noise-to-signal ratio $(\sigma_E^2/2^{-2t}\sigma_X^2)^{1/2}$ vs N. A theoretical curve (straight line) corresponding to (III-29) is shown. Since the white inputs used had real and imaginary parts uniformly distributed in $(-1, 1)$,† with $\sigma_x^2 = 2/3$, the theoretical curve used was

$$\left(\frac{\sigma_E^2}{2^{-2t}\sigma_X^2}\right)^{1/2} = (5/2)^{1/2}N^{1/2} \quad . \tag{III-30}$$

† Actually, a distribution in $(-1/\sqrt{2}, 1/\sqrt{2})$ should have been used, since the distribution which was used does not guarantee $|x(n)| < 1$, making it possible (though not probable) that an overflow can occur despite shifts at every array. However, such an overflow did not occur in any of our experimental runs.

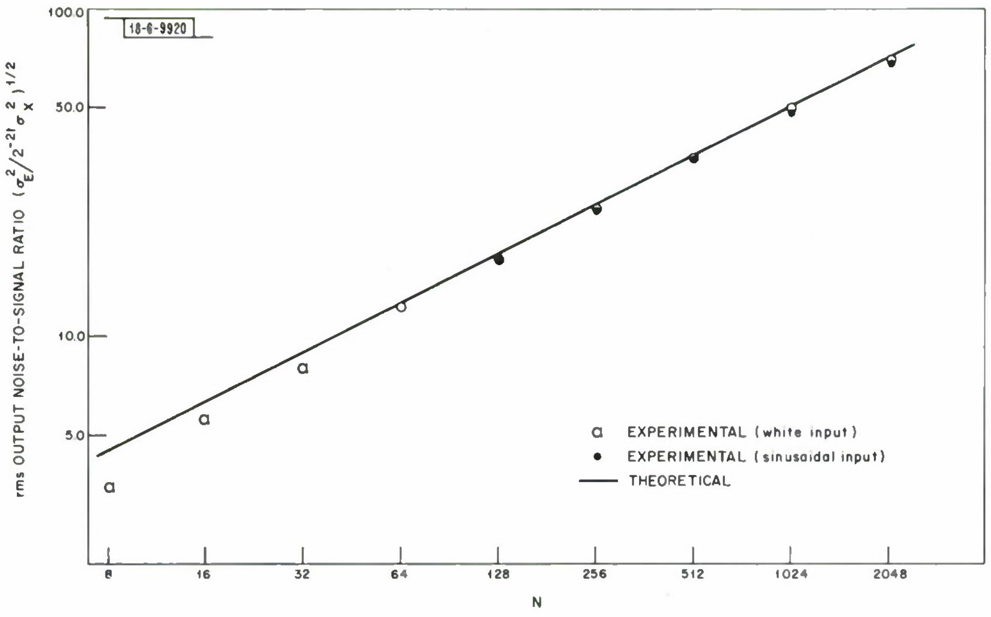


Fig. 12. Theoretical and experimental noise-to-signal ratios for fixed point FFT, with rounded arithmetic.

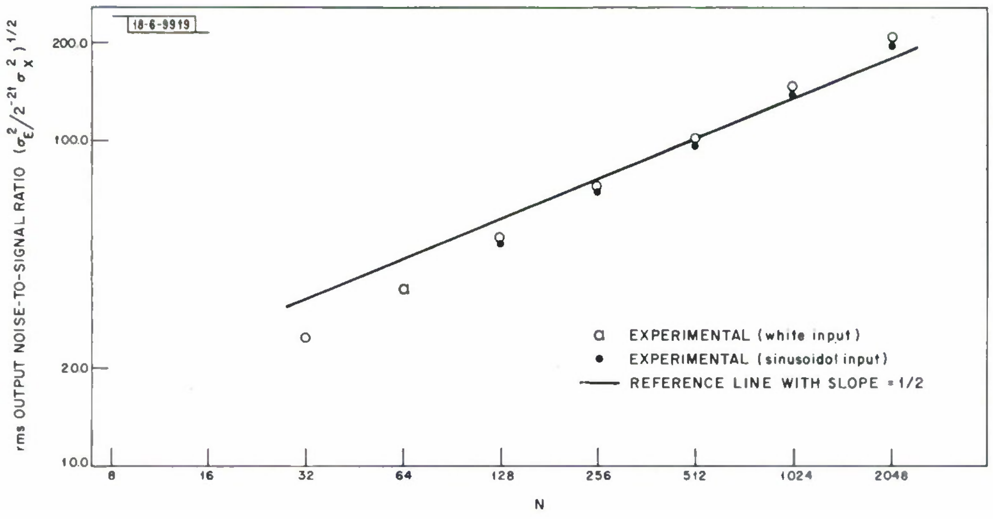


Fig. 13. Experimental noise-to-signal ratios for fixed point FFT, with truncation instead of rounding.

Experimental results for white noise inputs are plotted, where for each N the results for several white inputs were averaged to yield a stable output noise variance measurement. We also show experimental results for sinusoidal inputs, of the form $x(n) = \exp[j(2\pi qn/N) + \varphi]$. For the sinusoidal inputs, $\sigma_x^2 = 1$, so that the correct theoretical curve is

$$\left(\frac{\sigma_E^2}{2^{-2t}\sigma_X^2}\right)^{1/2} = (5/3)^{1/2} N^{1/2} \quad .$$

The actual experimental results were multiplied by $[(5/2)/(5/3)]^{1/2} = (3/2)^{1/2}$, so that they could be conveniently plotted alongside the already drawn theoretical curve of (III-30). For each N, the plotted results represent an average over several input frequencies.

For both white and sinusoidal inputs, the agreement of theory with experiment is excellent, and certainly justifies the "half-bit-per-stage" rule for the increase of noise-to-signal ratio. The fact that the experimental results lie consistently below the theoretical curve, more so for lower N, can be partially explained by the fact that multiplications by 1 and j introduce less roundoff noise than assumed in our model.

An issue of some practical importance is the question of how our results would change if truncation rather than rounding were used. This question has been investigated experimentally and the results are shown in Fig. 13. Although the actual values of noise-to-signal ratio are higher than for rounding, the slope of the curve remains essentially the same. This might be expected, since the half-bit-per-stage slope is determined chiefly by the signal attenuation (same for rounding and truncation) and not by the noise accumulation. The somewhat greater output noise for truncation is to be expected, due to the fact that a truncation noise source has a slightly higher mean-square than a rounding noise source, and perhaps also due to the correlation of truncation noise with signal sign.

When rounded arithmetic was used, except that the results of right shifts scaling the array were consistently rounded up (rather than rounded randomly up or down), it was found that the noise was somewhat higher than with randomized rounding, but lower than when truncation was used throughout the FFT.

C. ROUNDOFF NOISE - BLOCK FLOATING POINT

A block floating point FFT can be carried out in the following manner: the original array is normalized to the far left of the computer word, with the restriction that $|x(n)| < 1$; the computation proceeds in a fixed point manner, except that after every addition there is an overflow test; if overflow is detected, the entire array is shifted right one bit and the computation continues. The number of necessary shifts are counted to determine a scale factor or exponent for the entire final array. The output noise-to-signal ratio depends strongly on how many overflows occur, and at what stages of the FFT they occur. The positions and timing of overflows are determined by the signal being transformed, and thus, in order to analyze noise-to-signal ratio in block floating FFT, one needs to know the signal statistics. This is in contrast to the fixed point analysis above, where it was not necessary to assume specific signal statistics.

The necessary number of right shifts of the array is related to the peakiness of the DFT of the signal being transformed. If the constant signal $x(n) = 1$ or the single-frequency input $x(n) = \exp[j(2\pi/N)\, k_o n]$ is transformed, the output (with k_o an integer) will consist of a single nonzero point, and (for $N = 2^{\nu}$) ν scalings of the array will be necessary, one for each stage.

For these maximally peaked spectra, block floating point computation reduces to the fixed point computation described above. Generally, the more scalings of the array that are needed, the worse will be the output noise-to-signal ratio, so that the fixed point analysis of the preceding section (where scalings at every stage were included) can be considered to be a worst-case analysis for block floating point. In choosing a register length for a system (e.g., an FFT filter) using block floating point FFT, one would probably be wise to choose on the basis of a worst-case analysis. However, an overflow test and adaptive scaling procedure is usually simple to include in a computer program, and the valid question remains as to how much better than the worst case (fixed point) one might expect to do using a block floating point strategy. Here, we analyze block floating point FFT noise-to-signal ratio for the case of a white input signal. The DFT of a white signal is also white, and one might expect (since the spectral energy is spread, rather than peaked at a single point) that scalings at all the stages would not be necessary, and a noise-to-signal ratio advantage over fixed point would be gained.

A general scheme for analyzing this problem is as follows. First, we need to determine the probabilities of the various possible scaling patterns for a transform of a white signal. We refer to a scaling pattern by a notation such as SSSNSNSNS, meaning that scales (in a 2^9-point transform) take place at stages 1, 2, 3, 5, 7, and 9, with no scalings at the remaining stages. We might determine the probability, or frequency of occurrence, of given scaling patterns either theoretically or via an experimental histogram obtained by transforming many white sequences, and recording where the scales occur. For any particular scaling pattern, we could estimate the output noise-to-signal ratio by straightforward application of the techniques of the preceding section. The overall expected output noise-to-signal ratio for block floating point would be determined as an average of the noise-to-signal ratios for the various scaling patterns, weighted by the probability of these patterns.

Let us briefly illustrate a theoretical approach to finding probabilities of scaling patterns. Consider a white input sequence x(n), whose real and imaginary parts are each uniformly distributed in (−1, 1). We wish to find the probability of an overflow in the first array of computations. Since, in computing the first array (see Fig. 10), only additions and subtractions are involved, each random variable (real or imaginary) obtained is triangularly distributed in (−2, 2). An overflow will occur unless all these 2N variables lie inside (−1, 1). Thus, the probability of overflow is

$$P(\text{overflow}) = 1 - (3/4)^{2N}$$

since 3/4 is the probability that any one variable lies inside (−1, 1). Note that for any reasonably large value of N, P(overflow) for the first stage is very close to 1. In trying to pursue this kind of analysis through all the stages of, say, a 2^9-point FFT, the work quickly becomes quite cumbersome. In later stages, the distribution of the variables in the array is not so easily determined. Even if we assume that these variables become Gaussian, the task is still quite difficult due to the large number of possible scaling patterns which must be considered; for a ν-stage transform, we must consider 2^{ν} possible scaling patterns.[†]

[†] Here, we are assuming that a maximum of one shift is necessary at any given stage. This assumption would be true if scaling were controlled by the modulus of the complex numbers [see (III-19)]. But, normally, we test for overflow in the computation of real and imaginary parts, and in this case it is possible (though quite unlikely) to get up to two overflows in a stage, or a single overflow of up to 2 bits. We will ignore this possibility because it has not been observed in any of our experimental runs, and unduly complicates the analysis.

Because of these difficulties, we have resorted to experimental histograms to estimate the probabilities of the scaling patterns. Fortunately, it turns out that most of the 2^{ν} possible scaling patterns are highly improbable and can, for practical purposes, be neglected. For example, in FFT's of 500 different 2^9-point white sequences [Re and Im parts uniform in $(-1, 1)$], one of eight specific scaling patterns occurred in 499 of the cases. These patterns, together with their empirical probabilities, are

SSSNSNSNS	P = 0.24
SSSNSNSSN	P = 0.16
SSSNSSNNS	P = 0.15
SSSNSSNSN	P = 0.10
SSSSNNSNS	P = 0.13
SSSSNNSSN	P = 0.08
SSSSNSNNS	P = 0.08
SSSSNSNSN	P = 0.05

Note that in each case a total of 6 scalings are necessary, 3 fewer than for a fixed point computation. Thus, we would expect a noise-to-signal ratio improvement over that in the fixed point case.

We can show theoretically that, for these 512-point transforms, the probability of exactly 6 scales is very close to unity. Our initial array x(n) consists of 1024 (considering real and imaginary parts) uniformly distributed, independent random variables with variance $\sigma_{x_r}^2 = 1/3$. If no scalings were incorporated, the DFT X(k) would consist of 1024 approximately Gaussian (by the central-limit theorem), independent random variables with variance $\sigma_{X_r}^2 = 512/3$. Let $|X_r|_{\max}$ denote the magnitude of that variable (real or imaginary), in the DFT array, which has the largest magnitude. The probability that exactly 6 scalings will be needed is given by

$$P(\text{exactly 6 scalings}) = P(2^5 < |X_r|_{\max} < 2^6) \quad .$$

We have

$$P[|X_r|_{\max} < 2^6] = P[\text{all } |X_r| < 2^6] = \left[1 - 2Q\left(\frac{2^6}{\sqrt{512/3}}\right)\right]^{1024}$$

where

$$Q(\alpha) = \int_{\alpha}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\, dx \quad .$$

Evaluating this numerically, we obtain

$$P[|X_r|_{\max} < 2^6] \approx 0.99901398 \quad .$$

Similarly,

$$P[|X_r|_{\max} < 2^5] = \left[1 - 2Q\left(\frac{2^5}{\sqrt{512/3}}\right)\right]^{1024} \approx 3.9(10^{-7})$$

and thus

$$P[2^5 < |X_r|_{max} < 2^6] = P[|X_r|_{max} < 2^6] - P[|X_r|_{max} < 2^5]$$

$$= P[\text{exactly 6 scalings}] \approx 0.9990136 \quad .$$

Thus, we have verified theoretically that the probability of exactly 6 scalings is very close to unity. This technique can be used to estimate the total number of necessary scalings, for other values of N.

We can use empirical scaling pattern probabilities (such as those given above for $N = 2^9$) to predict average output noise-to-signal ratio. For any given scaling pattern, the output noise variance can be derived using the same techniques as for fixed point, with slight modifications. In computing the output noise variance at a particular output point due to a particular noise source which is connected to that output point. we multiply the noise source variance by $(1/2)^{2S}$, where S is the total number of scalings following the introduction of the noise source. The noise source variance itself will be σ_B^2 or $\sigma_{B'}^2$ (see preceding section), depending on whether or not the noise due to a scaling needs to be included. For any scaling pattern, the output signal variance is related to the input signal variance by

$$\sigma_X^2 = 2^{\nu} 2^{-2S_{total}} \sigma_x^2$$

where S_{total} is the total number of scalings. Once we have determined σ_E^2/σ_X^2 for all the reasonably probable scaling patterns, we determine an overall "theoretical" σ_E^2/σ_X^2 as an average over the various patterns of σ_E^2/σ_X^2 for a pattern, weighted by the probability of occurrence of this scaling pattern.

For values of N from 32 to 2048, experimental probabilities of scaling patterns have been obtained (for white inputs). By using these patterns and the procedure just described, "theoretical" output noise-to-signal ratios have been derived. The computations involved in deriving these results are straightforward but lengthy and are omitted. In the derivations, account was taken of the fact that multiplications by $\widetilde{W} = 1$ and j result in lower roundoff noise variance than for other values of $\widetilde{W}$ (although neglecting this fact changes the results very little). These theoretical results are represented as a curve (line segments joining the computed points) on Fig. 14. Also presented are experimentally measured values of output noise-to-signal ratio for block floating point FFT's of white input, using rounded arithmetic (which was assumed in the analysis). The experiments and theory agree closely. For comparison, a theoretical curve representing fixed point noise-to-signal ratio (for rounded arithmetic) is also shown. We see that for this kind of input (white), block floating point provides some advantages over fixed point, especially for the larger transforms. For N = 2048, the rms noise-to-signal ratio for block floating point is about 1/8 that of fixed point, representing a 3-bit improvement.

An experimental investigation was used to examine how the results for block floating point change, when truncation rather than rounding is used. The results of this experiment are also shown on Fig. 14. Noise-to-signal ratios are generally a bit or two worse than for rounding. The rate of increase of noise-to-signal ratio with N seems to be about the same as for rounding.

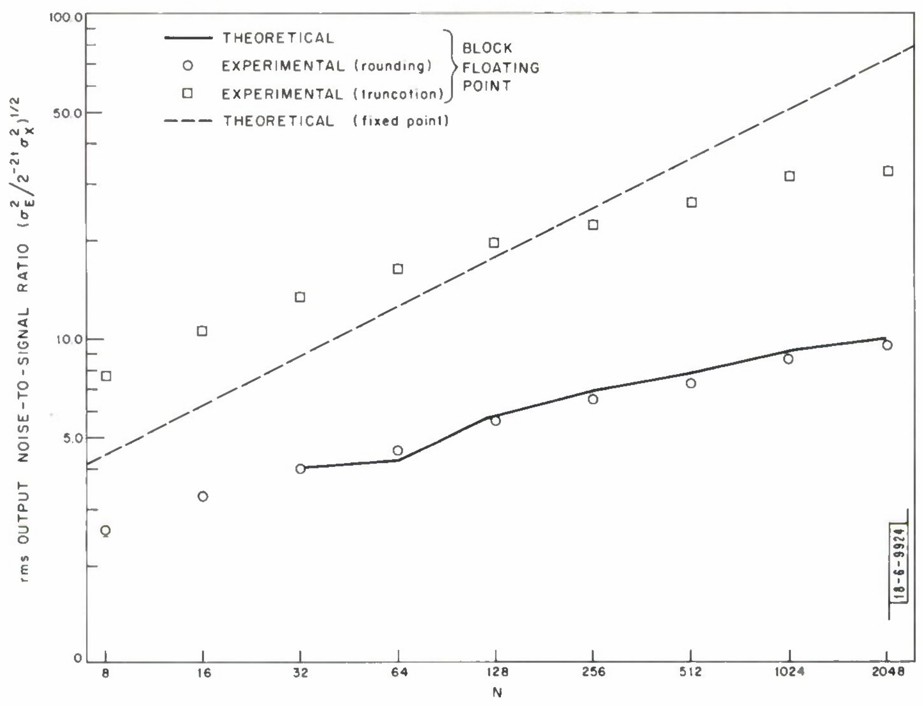


Fig. 14. Experimental and theoretical noise-to-signal ratios for block floating point FFT.

D. ROUNDOFF NOISE - FLOATING POINT

In this section, a statistical model for roundoff errors (as specified in Sec. I-C-2) is used to predict output noise-to-signal ratio when an FFT is computed using floating point arithmetic. The result, derived for the case of white input signal, is that the ratio of output noise variance to output signal variance is proportional to $\nu = \log_2 N$, where N is the number of points transformed. This predicted result is significantly lower than bounds previously derived[28] on mean-squared output noise-to-signal ratio, which are proportional to ν^2. That the statistical result increases more slowly with ν than the bound is certainly not surprising, since in a statistical analysis we assume that all noise sources in the system add as independent random variables, whereas, to obtain a deterministic bound, we must assume complete correlation between the noise sources.

The model applies to rounded arithmetic, and experimental noise measurements closely corroborate the theory when rounded arithmetic is used in the experiments. Even for nonwhite (e.g., sinusoidal) signals, experimental noise-to-signal ratios are observed to follow a linear dependence on ν. These experimental results seem to disagree with those obtained by Gentleman and Sande,[28] which indicated that the noise-to-signal ratio increased as ν^2 rather than as ν. Most of their experimental results pertained to truncation rather than rounding, and we have also verified that, for truncation, a quadratic dependence on ν fits the experimental data more closely than a linear dependence. In some of their experiments, however, rounding was used and a ν^2 dependence still was observed. But our rounding procedure included a random decision as to whether to round up or down, when a computed result lay midway between two machine numbers. We found that when this random rule was not used, but was replaced by an upward round of the mantissa in this midway situation, the experimental noise-to-signal ratios increased significantly, and could indeed be better fitted with a quadratic than with a linear curve.

This interesting observation is a possible explanation for the apparent discrepancy between our experimental results and those of Gentleman and Sande. The midway situation occurs quite frequently in floating point computation, especially for addition of numbers of the same order of magnitude, where the unrounded mantissas are very often only 1 bit longer than the rounded mantissas. Always rounding the mantissa up in this situation introduces correlation between roundoff error and signal sign, which contradicts the assumption that roundoff errors are signal-independent. This correlation seems to be enough to cause the output noise-to-signal ratio to increase significantly faster than ν.

1. Statistical Analysis

Our basic result for output noise-to-signal ratio in a floating point FFT of a white input signal is

$$\frac{\sigma_E^2}{\sigma_X^2} = 2\sigma_\epsilon^2 \nu \tag{III-31}$$

where σ_ϵ^2 is the variance of a floating point roundoff variable (see Sec. I-C-2). In the next few paragraphs, we give a rather heuristic derivation of this result.

Let us refer back to the FFT flow graph of Fig. 11. In the floating point case, as in the fixed point case, we can model the roundoff errors by introducing noise generators at all the nodes of the flow graph. Postulating that all roundoffs are independent of each other and of the signal, we assume that all the noise sources injected at the various nodes are mutually uncorrelated, and uncorrelated with the signal. Then, given the variances of the noise sources, our procedure for determining output noise variance is identical to the fixed point case. We draw a tree, such as that shown in Fig. 11, to indicate all the noise sources which affect a given output point. Then, since the transmissions on the arrows of the FFT flow graph all have unity magnitude, the resultant noise variance at the output point of interest is simply the sum of the variances of the noise sources injected at the nodes of the tree.

For the fixed point case, computing this resultant variance involved simply counting the nodes of the tree, since all noise sources had equal variance. But for the floating point case, all noise sources do not have equal variance, since the variance of a roundoff noise source is proportional to the variance of the signal being rounded, and the signal variance changes as we proceed from array to array in the FFT computation. This change is easily analyzed for the case of a white signal, where in passing from one stage to the next the signal remains white, and its variance simply doubles [this can be verified using (III-15)]. Thus, for white signals, the roundoff noise source variances are equal for a given array, but double as we pass from array to array.

The noise source variances double as we pass from stage to stage, but, as can be seen from the tree of Fig. 11, the number of noise sources affecting a given output point halves as we proceed forward from stage to stage. These effects exactly offset one another, so that the total contribution to output noise variance from the noise sources at any particular array does not vary with the position of the array. Since we have a total of ν arrays, the output noise variance is proportional to ν as indicated in (III-31).

To obtain the explicit result stated in (III-31), we need to find the noise variance at an output point, due to the noise sources at any particular array. Let us consider the first array (the array following the input array). If we assume a variance $\sigma_x^2 \equiv \mathcal{E}|x(n)|^2$ for the input, then we can show via our floating point roundoff model that the variance of the first array of noise sources is

$$\sigma_{u_1}^2 = 4\sigma_\epsilon^2 \sigma_x^2 \quad . \tag{III-32}$$

From the tree, we find that N/2, or $2^{\nu-1}$ of these noise sources affect any given output point, so that the resulting output noise variance is

$$\sigma_{E_1}^2 = 2^{\nu+1}\sigma_\epsilon^2\sigma_x^2 \quad . \tag{III-33}$$

Since there are ν arrays, all contributing equally to the output noise variance, the total output noise variance is

$$\sigma_E^2 = 2\sigma_\epsilon^2 \nu(2^\nu \sigma_x^2) \quad .$$

By noting that the output signal variance is related to the input signal variance by

$$\sigma_X^2 = 2^\nu \sigma_x^2$$

the result follows:

$$\frac{\sigma_E^2}{\sigma_X^2} = 2\sigma_\epsilon^2 \nu \quad . \qquad [(\text{III-31})]$$

A further result, which can be derived from our model, is an expression for the final expected output noise-to-signal ratio which results after performing an FFT and an inverse FFT on a white signal x(n). The inverse FFT introduces just as much roundoff noise as the FFT itself, and thus we can convince ourselves that the resulting output noise-to-signal ratio is

$$\frac{\sigma_E^2}{\sigma_x^2} = 4\sigma_\epsilon^2 \nu \tag{III-34a}$$

or just double the result in (III-31).

In order to see the implications of (III-31) or (III-34a) in terms of register length requirements, it is useful to express these results in units of bits. Corresponding to (III-31), we use

$$[\sigma_E^2/\sigma_X^2\sigma_\epsilon^2]_{\text{bits}} = 1/2 \log_2 (2\nu) \tag{III-34b}$$

to represent the number of bits by which the rms noise-to-signal ratio increases in passing through a floating point FFT. For example, for $\nu = 8$ this represents 2 bits, and for $\nu = 11$ it represents 2.23 bits. The number of bits of rms noise-to-signal ratio increases as $\log_2(\log_2 N)$, so that doubling the number of points in the FFT produces a very mild increase in output noise, significantly less than the half-bit-per-stage increase for fixed point computation. In fact, to obtain a half-bit increase in the result above, we would have to double ν, or square N.

A detail in our analysis which should probably be spelled out more carefully is the derivation of the expression (III-32) for the variance of the noise source introduced in a butterfly computation. In obtaining this result, we begin by expressing the butterfly computation (III-15a) in terms of real arithmetic [a similar analysis could be carried out for (III-15b)]

$$\mathrm{Re}\,[X_{m+1}(i)] = \mathrm{Re}\,[X_m(i)] + \mathrm{Re}\,\tilde{W}\,\mathrm{Re}\,[X_m(j)] - \mathrm{Im}\,\tilde{W}\,\mathrm{Im}\,[X_m(j)] \qquad \text{(III-35a)}$$

$$\mathrm{Im}\,[X_{m+1}(i)] = \mathrm{Im}\,[X_m(i)] + \mathrm{Re}\,\tilde{W}\,\mathrm{Im}\,[X_m(j)] + \mathrm{Im}\,\tilde{W}\,\mathrm{Re}\,[X_m(j)] \quad . \qquad \text{(III-35b)}$$

Because of roundoffs, we actually compute

$$fl\,[\mathrm{Re}\,X_{m+1}(i)] = \{\mathrm{Re}\,X_m(i) + [\mathrm{Re}\,\tilde{W}\,\mathrm{Re}\,X_m(j)\,(1+\epsilon_1)$$
$$- \mathrm{Im}\,\tilde{W}\,\mathrm{Im}\,X_m(j)\,(1+\epsilon_2)]\,(1+\epsilon_3)\}\,(1+\epsilon_4) \qquad \text{(III-36a)}$$

$$fl\,[\mathrm{Im}\,X_{m+1}(i)] = \{\mathrm{Im}\,X_m(i) + [\mathrm{Re}\,\tilde{W}\,\mathrm{Im}\,X_m(j)\,(1+\epsilon_5)$$
$$+ \mathrm{Im}\,\tilde{W}\,\mathrm{Re}\,X_m(j)\,(1+\epsilon_6)]\,(1+\epsilon_7)\}\,(1+\epsilon_8) \quad . \qquad \text{(III-36b)}$$

An expression for the noise $U_m(i)$ introduced by roundoff is obtained by subtracting (III-35) from (III-36) and neglecting terms of second or higher order in the roundoff variables ϵ_i. We obtain

$$U_m(i) = \mathrm{Re}\,[X_m(i)]\,(\epsilon_4) + \mathrm{Re}\,\tilde{W}\,\mathrm{Re}\,[X_m(j)]\,(\epsilon_1+\epsilon_3+\epsilon_4) - \mathrm{Im}\,\tilde{W}\,\mathrm{Im}\,[X_m(j)]$$
$$\times(\epsilon_2+\epsilon_3+\epsilon_4) + j\,\{\mathrm{Im}\,[X_m(i)]\,(\epsilon_8) + \mathrm{Re}\,\tilde{W}\,\mathrm{Im}\,[X_m(j)]$$
$$\times(\epsilon_5+\epsilon_7+\epsilon_8) + \mathrm{Im}\,\tilde{W}\,\mathrm{Re}\,[X_m(j)]\,(\epsilon_6+\epsilon_7+\epsilon_8)\} \quad . \qquad \text{(III-37)}$$

For a white input signal, $\mathrm{Re}\,[X_m(i)]$, $\mathrm{Re}\,[X_m(j)]$, $\mathrm{Im}\,[X_m(i)]$, and $\mathrm{Im}\,[X_m(j)]$ are mutually uncorrelated with equal variances, and we can obtain the noise source variance as

$$\sigma_u^2 \equiv \mathcal{E}|U_m(i)|^2 = 4\sigma_\epsilon^2\,\mathcal{E}|X_m(i)|^2 \quad . \qquad \text{(III-38)}$$

For the computation of the first array, $\mathcal{E}|X_m(i)|^2 = \mathcal{E}|X_o(i)|^2 = \mathcal{E}|X(n)|^2 = \sigma_x^2$, and (III-32) follows.

In this analysis, we have not considered the fact that a reduced error variance is introduced when $\tilde{W} = 1$ or j. In fact, for $\tilde{W} = 1$, since multiplications by 1 can be performed noiselessly, we have $\epsilon_1 = \epsilon_2 = \epsilon_3 = \epsilon_5 = \epsilon_6 = \epsilon_7 = 0$ in (III-36), and the noise source variance is half that given in (III-38). A similar result holds for $\tilde{W} = j$. For a specified radix 2 algorithm, such as the decimation in time algorithm shown in Fig. 10, these reduced variances for $\tilde{W} = 1$ and j can be included in the model to obtain a slightly reduced prediction for output noise-to-signal ratio. However, for reasonably large N, this modified noise analysis yields only slightly better predictions of output noise than does the simplified analysis above.

A consequence of our analysis leading to (III-31) is that the output noise is white. This follows from the fact that each array of noise sources is white.† The reduced noise source variance for $\tilde{W} = 1$ and j implies that for some arrays there will be a variation of noise source variance over the array. This implies a slight variation of output noise variance over the output array, and thus our modified noise analysis will only predict an average noise variance over the output

† Actually, as in the fixed point case, the pair of noise sources introduced in computing the two outputs of a given butterfly are correlated. This means that there is a slight correlation of noise over the output array, but does not affect our result for output noise variance since the top and bottom output nodes of a butterfly affect a disjoint set of points in the output array.

array. For the decimation in time algorithm depicted in Fig. 10 (which was the algorithm used in our experimental work), such a modified analysis was carried out. For this case, only $\widetilde{W} = 1$ is used in the first array of computations, and only $\widetilde{W} = 1$ and j are used in computing array 2 from array 1. In computing array 3 from array 2, half the butterflies involve $\widetilde{W} = 1$ or j; in the next array, one-quarter of the butterflies involve $\widetilde{W} = 1$ or j; etc. The result of our analysis, which is to be interpreted as an average (over the output array) of noise to signal variance is

$$\frac{\sigma_E^2}{\sigma_X^2} = 2\sigma_\epsilon^2 \, [\upsilon - (3/2) + (1/2)^{\upsilon-1}] \quad . \tag{III-39}$$

As υ becomes moderately large (say, $\upsilon \geqslant 6$), we see that (III-39) and (III-31) predict essentially the same linear rate of increase of σ_E^2/σ_X^2 with υ.

2. Experimental Noise Measurements

The essential result of the above analysis is that mean-squared output noise-to-signal ratio in a floating point FFT is proportional to υ. Experimental testing of this prediction was deemed particularly important, both because of the many assumptions inherent in the model, and because the prediction seemed to be contradicted by the experimental results of Gentleman and Sande, who observed (for white noise input) that measured output noise-to-signal ratios were approximately proportional to υ^2. (Their noise measurements were carried out by performing a transform and inverse, and subtracting the initial and final sequences.) Our first thought at an explanation of the discrepancy between their results and ours was that most of their experiments were carried out using truncation, whereas our model applies to rounded arithmetic. However, a few of their experiments used rounding, and even there a υ^2 dependence of mean-squared output noise-to-signal ratio was observed.

We carried out experimental noise measurements as follows. To check (III-31), a white sequence (consisting of uniformly distributed random variables) was generated and transformed twice – once using rounded arithmetic with a short (e.g., 12-bit) mantissa, and once using a much longer (27-bit) mantissa. The results were subtracted, squared, and averaged to estimate the noise variance. For each $N = 2^\upsilon$, this process was repeated for several white noise inputs to obtain a stable estimate of roundoff noise variance. To check (III-34a), white sequences were put through an FFT and inverse (using, for example, 12-bit rounded arithmetic), and the mean-squared difference between initial and final sequences was taken.

The initial results we obtained seemed to be more in agreement with Gentleman and Sande's experiments than with our theory, since the measured noise-to-signal ratios were higher than we predicted and could be fitted more closely with a curve of the form $a\upsilon^2$ than with a linear curve in υ. The explanation for this seemed to be that our model of independent noise generators must be incorrect, and a significant correlation between signals and floating point roundoff errors must exist. However, the rounding procedure used in these initial experiments did not include a random choice as to whether to round up or down when a result lay midway between two machine numbers. We merely looked at the first extra bit of the (unsigned) mantissa, added 2^{-t} to the mantissa if this bit were 1, and otherwise simply truncated to t-bits. When we modified this procedure, and rounded randomly up or down in the midway situation (where the first extra bit of the mantissa is 1, and all remaining extra bits are 0), the experimental noise-to-signal ratios followed our predictions quite closely. It turns out that, in floating point addition of two numbers of the same order of magnitude, this midway situation occurs quite frequently.

Rounding always up, rather than randomly up or down, in this situation introduces a correlation between roundoff error and signal sign.

Our results for white input signals are summarized in Figs. 15 and 16. Figure 15 shows the results for an FFT, and Fig. 16 displays our results for an FFT and inverse. The figures show straight lines corresponding to the simple theoretical results (III-31) and (III-34a); also shown are modified theoretical curves corresponding to the analysis leading to (III-39). Experimental results corresponding both to our initial experiments (nonrandomized rounding) and to our experiments using randomized rounding are presented. The results for randomized rounding correspond very closely with our modified noise analysis, and reasonably closely with our simpler analysis. The results for nonrandomized rounding do not follow our theory very well, and can better be fitted by the quadratic curves shown. Actually, when calibrated in terms of bits, the results for randomized and nonrandomized rounding are not that different since a factor of 4 on Figs. 15 or 16 corresponds to 1-bit difference in rms noise-to-signal ratio. For the plotting of theoretical curves, the empirical value

$$\sigma_\epsilon^2 = (0.21)2^{-2t} \qquad \text{(III-40)}$$

was used for the variance of the roundoff variables. This is very close to the value (II-48) which was found empirically to fit noise measurements on simple recursive filters.

For the case of nonrandomized rounding, experiments were performed to isolate the effects of error due to rounding of multiplications and additions. For example, to measure the total

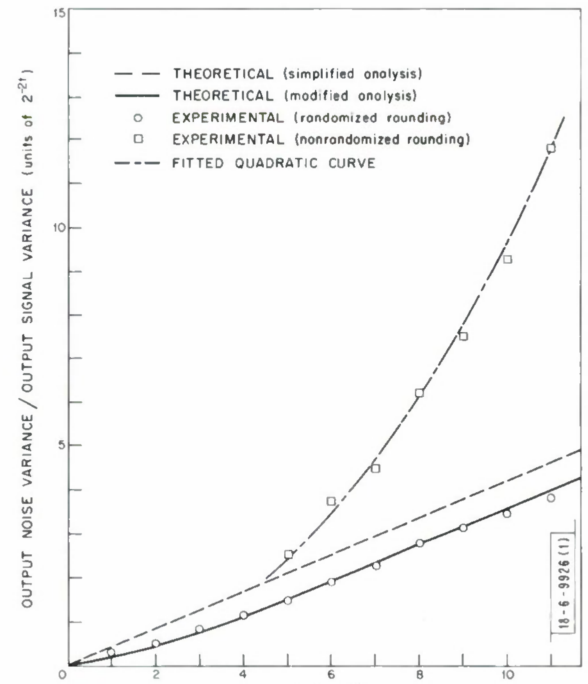


Fig. 15. Experimental and theoretical noise-to-signal ratios for floating point FFT.

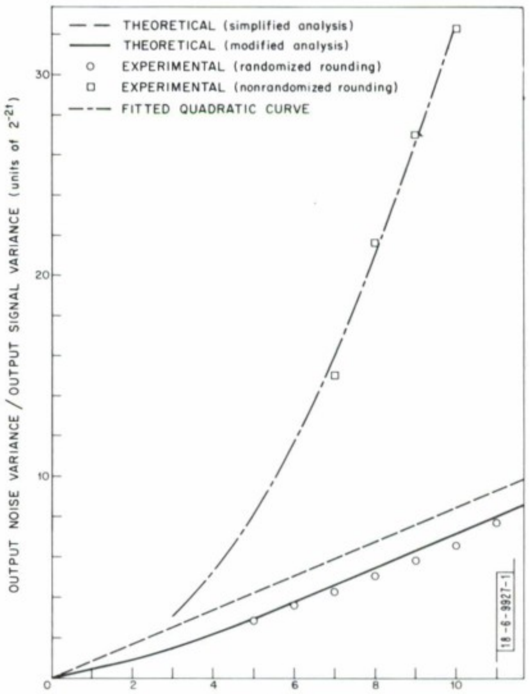


Fig. 16. Experimental and theoretical noise-to-signal ratios for floating point and inverse FFT's.

noise due just to additions, all multiplications were rounded to 27 bits, while additions were rounded to 12 bits. We found that the output noise-to-signal ratio for this case (of inaccurate additions and accurate multiplications) increased faster than υ, and was almost as large as when additions and multiplications were both performed inaccurately. The output noise-to-signal ratio, measured with inaccurate multiplications and accurate additions, was much lower, and increased essentially linearly with υ. These results supported our conjecture that the signal-noise correlation was introduced mainly in the rounding of additions.

Noise-to-signal ratios were also measured for the case where truncation, rather than rounding, was used in the arithmetic. As might have been expected, the measured ratios were significantly higher than in the case of rounding, and were observed to increase more than linearly with υ. The results are shown in Fig. 17 for both a transform, and a transform and inverse. For the latter case, the experimental values of $\sigma_E^2/2^{-2t}\sigma_x^2$ were halved for convenient plotting. Quadratic curves of the form $a\upsilon^2$ have been fitted to the experimental results. We note that the measured values for transform and inverse were somewhat more than double the results for a single transform.

At this point, an important comment should be made concerning the fitted quadratic curves of Figs. 15 to 17. These curves represent merely an empirical fit to the data, and do not correspond to any theoretical result which we have been able to derive. They are drawn to indicate that the form of our experimental data does not disagree with that of Gentleman and Sande. However, many other curves could be fitted to the data, and the fact that the bound on output noise-to-signal ratio is proportional to υ^2 does not imply that, if we could model theoretically the correlation between signal and roundoff error, the predicted noise-to-signal ratio would be proportional

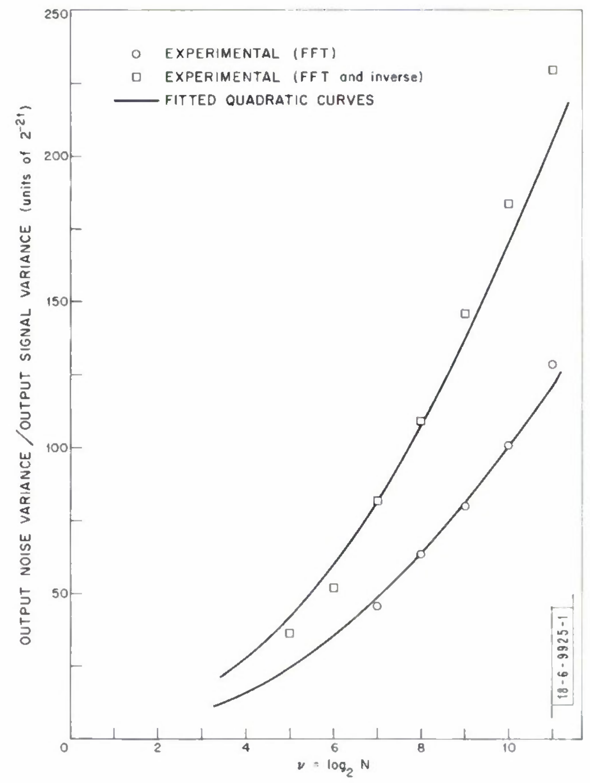


Fig. 17. Experimental noise-to-signal ratios for floating point FFT and FFT – inverse FFT; truncation used instead of rounding.

to ν^2. In the bounding analysis, complete correlation between noise sources is assumed, whereas (for nonrandomized rounding or truncation) we are dealing with only slight correlation. The experimental results are well below Gentleman and Sande's deterministic bounds on output noise-to-signal ratio.

Our analysis, and all the above experiments, applied to the case of white signal. Some experimental investigation has been carried out as to whether our predictions are anywhere near valid when the signal is nonwhite. Specifically, the noise introduced in computing an FFT was measured for sinusoidal signals of several frequencies, for $\nu = 8$, 9, 10, and 11. The results, averaged over the input frequencies used, were within 15 percent of those predicted by (III-39). In these experiments, our "randomized" rounding procedure was used.

3. FFT vs Direct Fourier Transform Accuracy

An issue of some interest is the question of how the accuracy obtained in an FFT computation compares with that obtained in computing a DFT by "straightforward" means, i.e., direct summing of products as implied by the definition (III-2) of the DFT. Gentleman and Sande[28] studied this problem for the floating point case, obtaining (for both the direct and FFT methods) deterministic bounds on the ratio of the rms output error to the rms output signal. The bound they obtained for the direct method increases much faster with N than the corresponding bound for the FFT.

A quantitative statement of their results is as follows. Consider an N-point sequence x(n). First, suppose we compute its DFT and then its inverse DFT by direct summing of products, obtaining a result x(n) + e(n), where e(n) is the error due to roundoffs. Then,

$$\frac{\sqrt{\frac{1}{N}\sum_{n=0}^{n-1}|e(n)|^2}}{\sqrt{\frac{1}{N}\sum_{n=0}^{n-1}|x(n)|^2}} \equiv \frac{e_{rms}}{x_{rms}} \leqslant (2)(1.06)2^{-t}(2N)^{3/2} \quad . \tag{III-41}$$

(Similarly, a bound could be derived for a single transform, with no inverse.)

Now, suppose we compute the DFT and inverse by means of a radix 2 FFT algorithm, where $N = 2^\nu$. The bound on rms output error ÷ rms output signal becomes

$$\frac{e_{rms}}{x_{rms}} \leqslant (2)(1.06)2^{-t}(2)^{3/2}\nu \quad . \tag{III-42}$$

For large N, (III-42) is a much lower bound than (III-41). It is of interest to compare the bound (III-42) with our corresponding result (III-34a), which was obtained via a statistical analysis. Stated in rms terms, our statistical result is

$$\frac{\sigma_E}{\sigma_x} = (2)(0.21)^{1/2}2^{-t}\nu^{1/2} \quad . \tag{III-43}$$

Our result is proportional to $\nu^{1/2}$ rather than ν, because in the statistical analysis it was assumed that errors from the different stages of the FFT added as independent random variables. In the bounding analysis leading to (III-42), it was necessary to assume that the errors from different stages add as completely correlated noises, and in the worst possible way.

In deriving their bound (III-41) for the direct Fourier transform case, Gentleman and Sande assume that the summing of products is performed in a cumulative way – that is, the $(n+1)^{st}$ product is added to the sum of the first n products, and so on. However, a slightly different, more error-free technique could be used for summing the products. Suppose instead, we were to sum (III-2) in a treelike fashion; that is, the N products were summed in pairs, the N/2 results summed in pairs, etc. Then, for a DFT and inverse, the bound corresponding to (III-41) is

$$\frac{e_{rms}}{x_{rms}} \leqslant 2(1.06)2^{-t}(\nu + 1)\sqrt{2N} \quad . \tag{III-43}$$

This bound is lower than (III-41), but still increases faster with N than the bound (III-42) for the FFT. A qualitative explanation for this latter fact is as follows. In the tree-sum computation, all output points are computed independently, so that in deriving a bound we must assume that all errors in the output array assume their maximum possible values. In the FFT computation, the output points are not computed independently. If the effects of roundoff errors reinforce each other in the worst possible way at a particular output point, they will tend to cancel at another output point.

In the above discussion, we have considered deterministic bounds, rather than a statistical analysis of the output error. A statistical analysis of roundoff errors incurred in implementing the DFT, using the treelike summation technique, predicts a linear dependence of the rms output noise-to-signal ratio on $\nu^{1/2}$, similar to (III-43). Thus, using a statistical analysis, we would conclude that the accuracy of a direct "tree-summed" Fourier transform is essentially the same as the accuracy of an FFT.

We should note that, compared with cumulative summing, the treelike summation requires some additional memory (to store partial sums), and perhaps more inconvenient indexing. However, for any reasonably large N, this issue is academic since, for reasons of increased speed, we would use the FFT (and neither cumulative nor treelike summation) to perform the computation.

E. SUMMARY OF FFT QUANTIZATION EFFECTS

Let us summarize and comment on our main results concerning quantization effects in the FFT.

In Sec. III-A, we studied the effect of coefficient quantization. We represented the error due to coefficient quantization as the result of multiplication of the input sequence by a "stray" matrix determined by the coefficient errors. To analyze the situation further, we needed to make the fairly crude assumption that all elements of the stray matrix were uncorrelated. Using this assumption, we obtained the result (for fixed point, rounded coefficients)

$$\left.\frac{\sigma_E^2}{\sigma_X^2}\right)_{\text{coefficient errors}} = (\nu/6)2^{-2t} \tag{III-44}$$

for the mean-squared output error-to-signal ratio. For floating point, rounded coefficients, essentially the same result (except for a multiplicative constant) was obtained. The key conclusion is that this error-to-signal ratio increases as $\nu = \log_2 N$, and experimental results (as displayed in Fig. 9) were in reasonable agreement with this conclusion. This result can help us estimate how many bits of accuracy are needed in our coefficient table, for a given requirement on output error-to-signal ratio. In general, the number of bits retained in the coefficients need not be equal to the number of bits retained in the rounding of multiplications. For example, in fixed point, the ratio of the variance of output noise due to roundoffs, to output signal variance, is proportional to N rather than to $\log_2 N$ as in (III-44). Thus, for large N, it would be reasonable to retain fewer bits in the coefficients than in the rounding of multiplications.

In Sec. III-B, we analyzed the effects of roundoff errors and the overflow constraint to obtain formulas for output noise-to-signal ratio in a fixed point FFT. We found that to guarantee against overflow (using preset attenuators, and no overflow test) and maintain the lowest possible noise-to-signal ratio, it was desirable to keep the input array as large as possible [under the constraint $|x(n)| < 1$] and incorporate attenuations of 1/2 at each stage. If this method were used, the resulting prediction for output noise-to-signal ratio (for rounded arithmetic) was

$$\left.\frac{\sigma_E^2}{\sigma_X^2}\right)_{\text{roundoff, fixed point}} = \frac{(5/3)N2^{-2t}}{\sigma_x^2} \quad . \tag{III-45}$$

Our analysis also yielded the result that the output noise sequence E(k) is essentially white. An implication of the result (III-45) is that if we double N, adding another stage to the FFT, the rms output noise-to-signal ratio increases by half a bit. We validated (III-45) experimentally, for both white and sinusoidal input sequences. We investigated experimentally the question of how our results would change if truncation rather than rounding were used, and found that, although the actual values of noise-to-signal ratio were higher for truncation, the half-bit-per-stage slope was retained. This might have been expected since this slope [the proportionality to N in (III-45)] is determined chiefly by the signal attenuation (same for rounding and truncation) and not by the noise accumulation.

Using (III-45) in conjunction with (III-44), we could specify register length requirements for a multiplier to be used in a fixed point FFT. To illustrate how this might be done, let us consider a specific example. Suppose the longest FFT we wished to compute was $N = 2^{10} = 1024$. Then, from (III-44), we obtain

$$\begin{aligned} 1/2 \log_2 (\sigma_E^2/\sigma_X^2)_{\text{coefficient}} &= \log_2 (5/3)^{1/2} - t_c \\ &= 0.37 - t_c \end{aligned} \tag{III-46}$$

and, from (III-45), assuming $\sigma_x^2 = 1/3$ (as for white input),

$$\begin{aligned} 1/2 \log_2 (\sigma_E^2/\sigma_X^2)_{\text{roundoff}} &= 1/2 \log_2 [(5)(2^{10})2^{-2t_r}] \\ &= 6.16 - t_r \end{aligned} \tag{III-47}$$

where t_c and t_r represent the number of bits retained (not including sign) in the coefficients and rounded results, respectively. Suppose we require that neither the error-to-signal ratio due to coefficient quantization nor the noise-to-signal ratio due to roundoff exceeds −60 dB. This implies that both (III-46) and (III-47) be less than −10 bits, so that the minimum register lengths are $t_c = 11$ bits and $t_r = 17$ bits. Thus, if we include the sign bits, a 12×18-bit multiplier would be required — 12 bits for the coefficient, and 18 bits for the signal. An 18-bit rounded output would be needed from each multiply.

In Sec. III-C, we studied the effects of roundoff noise in a block floating point FFT, where the array was scaled by overflow tests and right shifts. We observed that fixed point computation, where scalings are imposed at every stage, is a worst case (with respect to noise-to-signal ratio) for block floating point. We found that for inputs with a fairly flat spectrum, significant improvements over the fixed point case could be attained in output noise-to-signal ratio. Detailed analysis and experimentation were carried out for the case of white input signal.

In Sec. III-D, a statistical model for roundoff errors was used to predict output noise-to-signal ratio in a floating point FFT. The result, derived for the case of white input signal, was

$$\left.\frac{\sigma_E^2}{\sigma_X^2}\right)_{\text{roundoff, floating point}} = (0.42)\nu 2^{-2t} \quad . \qquad \text{(III-48)}$$

This predicted result was significantly lower than bounds previously derived on mean-squared output noise-to-signal ratio, which were proportional to ν^2. The result was verified experimentally when "randomized rounding" was used. When either "nonrandomized rounding" or truncation was used, a greater than linear increase with ν of the output noise-to-signal ratio was observed. This was attributed to the fact that these procedures introduced a correlation between the rounding errors and the rounded quantities. An implication of (III-48) and (III-14) [the floating point counterpart of (III-44)] is that, for floating point, the noise-to-signal ratios due to both coefficients and roundoffs are proportional to $\nu 2^{-2t}$, so that it seems reasonable to retain the same number of bits for the coefficients as for rounding.

We concluded Sec. III with a discussion of the relative accuracy of the FFT (floating point) and a direct Fourier transform. It was pointed out that if a treelike summation is used to carry out the direct Fourier transform, then (on a statistical basis) its accuracy is essentially the same as that of the FFT.

IV. FREQUENCY SAMPLING FILTERS

We now study quantization effects in frequency sampling filters.[2,4] The term "frequency sampling filter" is used here to refer to a particular realization of a finite duration impulse response filter, by means of a comb-filter and a bank of resonators. The terminology is a bit misleading, since "frequency sampling" is actually more descriptive of the design procedure used than of the particular implementation. Once designed, the filter could be realized in a variety of ways, including fast convolution (using the FFT). In fact, an important objective of this section and Sec. V is to compare, on the basis of differing quantization effects, recursive and FFT realizations of finite duration impulse response filters.

Gold and Jordan[32] have shown that any finite duration impulse response filter can be realized in the frequency sampling form, that is by means of a comb-filter and a bank of resonators. Consider a filter whose impulse response h(n) is time-limited to N samples. Its transfer function is

$$H^*(z) = \sum_{n=0}^{N-1} h(n)\, z^{-n} \quad . \tag{IV-1}$$

But the finite duration h(n) can be expressed in terms of its DFT, H(k), as

$$h(n) = \frac{1}{N} \sum_{k=0}^{N-1} H(k) \exp\,[j(2\pi/N)\, nk] \quad . \tag{IV-2}$$

The H(k) represent N samples, uniformly spaced around the unit circle, of the z-transform H(z). Using (IV-2) in (IV-1), we can express $H^*(z)$ in terms of its frequency samples as

$$H^*(z) = \frac{1 - z^{-N}}{N} \sum_{k=0}^{N-1} \frac{H(k)}{1 - e^{j2\pi k/N} z^{-1}} \quad . \tag{IV-3}$$

This is basically the frequency sampling realization. The term $(1 - z^{-N})$ represents a comb-filter, which has N zeros equally spaced around the unit circle. The sum represents a bank of single pole filters. Each pole cancels one of the zeros of the comb-filter, so that the composite filter has only zeros. The continuous frequency response $H^*(e^{j\omega})$ passes exactly through its N samples H(k) according to

$$H^*(e^{j\omega})\big|_{\omega=2\pi k/N} = H(k) \quad . \tag{IV-4}$$

If we confine attention to the case where the H(k) are real, and H(k) = H(N − k), we can combine complex conjugate terms in (IV-3) to obtain

$$H^*(z) = \left(\frac{1 - z^{-N}}{N}\right) \left\{ \frac{H(0)}{1 - z^{-1}} + \sum_{k=1}^{[N/2]} \frac{2H(k)\,[1 - \cos(2\pi k/N)\, z^{-1}]}{1 - 2\cos(2\pi k/N)\, z^{-1} + z^{-2}} \right\} \tag{IV-5}$$

where

$$[N/2] = \begin{cases} N/2 & \text{N even} \\ (N-1)/2 & \text{N odd} \quad . \end{cases}$$

This filter can be realized using resonators with real coefficients, and has linear phase.[†] The filter can be designed by specifying the frequency samples H(k), or by any of the usual methods for designing nonrecursive filters. Usually, we would choose the H(k) to alternate in sign so that the continuous magnitude response $|H^*(e^{j\omega})|$ will smoothly interpolate between the sample values. Excellent filter designs (band- or low-pass filters with low sidelobes and linear phase) corresponding to the form (IV-5) can be obtained.

However, the quantization problems involved in implementing (IV-5) are quite severe, basically due to the need for pole-zero cancellations on the unit circle. Our aim here is to study these quantization effects so that register length requirements can be decided upon. In Sec. V, we shall consider the quantization problems which occur when the same filter designs are implemented via the route of the FFT.

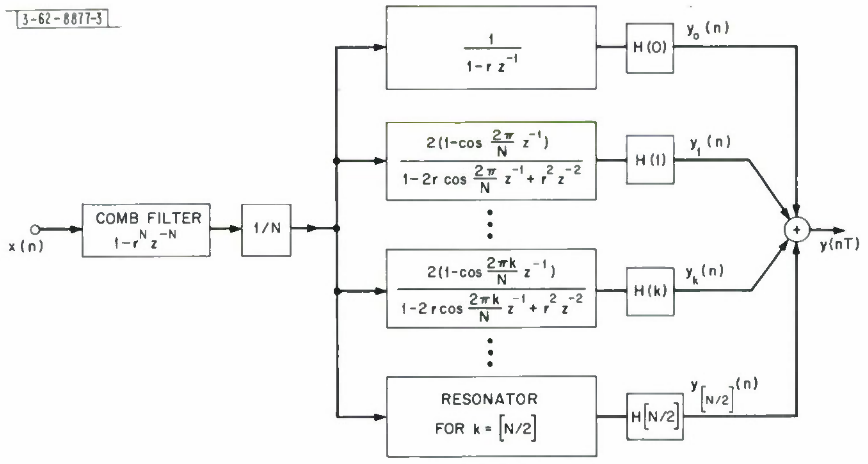
In practice, we would not attempt to realize exactly the unit circle pole-zero cancellations implied by (IV-5), but would move the poles of the resonators and the zeros of the comb-filter slightly inside the unit circle, so that

$$H^*(z) = \frac{1-r^N z^{-N}}{N}\left\{\frac{H(0)}{1-rz^{-1}} + \sum_{k=1}^{[N/2]} \frac{2H(k)\,[1-\cos(2\pi k/N)\,z^{-1}]}{1-2r\cos(2\pi k/N)\,z^{-1}+r^2z^{-2}}\right\} \qquad \text{(IV-6)}$$

where r is chosen slightly less than 1, say,

$$r = 1-\delta \quad . \qquad \text{(IV-7)}$$

A block diagram of the frequency sampling filter is shown in Fig. 18. The distance δ of the poles from the unit circle is an important parameter which must be chosen by the designer. As



3-62-8877-3
x(n)
COMB FILTER
$1-r^N z^{-N}$
1/N
$\frac{1}{1-r z^{-1}}$
$\frac{2(1-\cos\frac{2\pi}{N} z^{-1})}{1-2r\cos\frac{2\pi}{N} z^{-1} + r^2 z^{-2}}$
$\frac{2(1-\cos\frac{2\pi k}{N} z^{-1})}{1-2r\cos\frac{2\pi k}{N} z^{-1} + r^2 z^{-2}}$
RESONATOR FOR k = [N/2]
H(0)
H(1)
H(k)
H[N/2]
$y_0(n)$
$y_1(n)$
$y_k(n)$
$y_{[N/2]}(n)$
+
y(nT)

Fig. 18. Block diagram of frequency sampling filter.

we decrease δ, both roundoff noise and coefficient quantization problems become more severe. But as δ is made larger, the filter characteristics begin to deviate more and more from those of the original frequency sampling filter, which is generally designed assuming $\delta = 0$.

[†]Actually, precise linear phase will be obtained if N is odd. If N is even, only approximate linear phase will be obtained, unless h(0) = 0, in which case precise linear phase is obtained.

A. COEFFICIENT QUANTIZATION†

Coefficient quantization causes the pole pair of each resonator in our frequency sampling filter to deviate in position in the z-plane from the corresponding zeros of the comb-filter. Since the poles and zeros are very close to the unit circle, the imperfect pole-zero cancellations can cause severe ripples in the filter's frequency response. This can be compensated for somewhat by increasing the parameter δ, to move the poles and zeros further inside the unit circle. However, making δ too large will also cause the frequency response to change appreciably from that of the original frequency sampling filter, which is generally designed assuming $\delta = 0$. The expected deviation in pole positions clearly increases with decreased register length, so that one's choice of δ should depend on the register length. Also, the expected deviation in pole positions depends on the particular form of resonator chosen (e.g., coupled or direct). When resonators of low resonant frequency are required, the lower coefficient sensitivity of the coupled form might induce one to choose this form, despite the added computation needed in its operation.

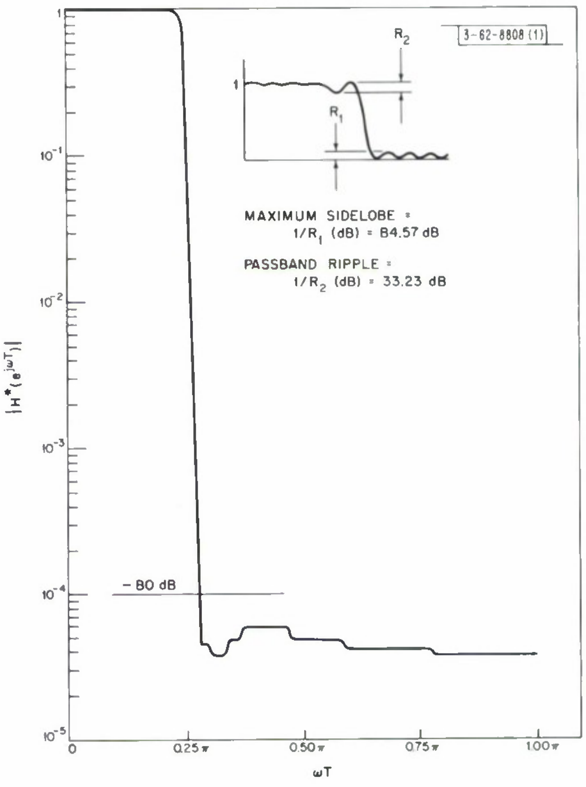


Fig. 19. Frequency response of low-pass frequency sampling filter.

As we see from this discussion, the key problem here is one we have discussed in detail in a preceding section, namely the problem of coefficient quantization in digital resonators. Rather than repeat any analysis, we shall illustrate here by example the composite effects of coefficient quantization on a complicated frequency sampling filter. We focus on the case of fixed point coefficients, making only a brief qualitative comment as to how we might expect the floating point case to compare with the fixed point case.

A low-pass filter design, suitable for implementation via Fig. 18 [or (IV-6)], was borrowed from Jordan and Gold.[33] Its specifications are: $H(k) = (-1)^k$, $k = 0, 1, \ldots, 31$; $H(32) = 0.7$; $H(33) = -0.225$; $H(34) = 0.01995$; $H(k) = 0$, $k = 35, 36, \ldots, N/2$; $H(k) = H(N-k)$; $N = 256$. Its frequency response $|H^*(e^{j\omega T})|$ (we recall that we are assuming for convenience that $T = 1$) is shown in Fig. 19. The maximum peak-to-peak passband ripple, as defined in the figure, is 33.23 dB down from the DC gain, and the maximum sidelobe is 84.57 dB down from the DC gain. Only the envelope of the sidelobes is shown.

The effects of coefficient quantization were examined as follows. A value of δ was chosen, and the filter coefficients were computed and quantized to a chosen number of bits (18 or less). The filter, with these quantized coefficients, was pulsed, using 36-bit arithmetic to minimize the effect of roundoff noise, and the impulse response thus obtained was Fourier transformed

† Most of the results reported here have been previously reported by the author.[12]

(also 36-bit arithmetic) to find the frequency response. With quantized coefficients, the impulse responses did not exactly terminate after 256 samples. However, it was observed that, after 2048 samples, the impulse response was zero or negligible, and this was the number of points used in computing the DFT.

For the first experiment, the resonators were programmed using the direct form recursion

$$y_k(n) = -r^2 y_k(n-2) + 2r \cos \frac{2\pi k}{N} y_k(n-1) - \cos \frac{2\pi k}{N} w(n-1) + w(n) \qquad \text{(IV-8)}$$

for the kth resonator. The comb-filter was programmed as

$$Nw(n) = x(n) - r^N x(n-N) \quad . \qquad \text{(IV-9)}$$

The first half of Table II shows the performance of this filter as a function of register length used for the coefficients (sign included), compared with the 36-bit "ideal" performance. For each register length, a range of values of δ was tried in order to decide upon that value which yielded performance as close as possible to the ideal. The criteria used for the search

TABLE II

PERFORMANCE OF FREQUENCY SAMPLING FILTER WITH QUANTIZED COEFFICIENTS

Register Length	δ "opt"	Maximum Sidelobe* (dB)	Maximum Passband Ripple† (dB)
Performance with Direct Form Resonators			
36	2^{-17}	84.57	33.23
18	2^{-9}	82.63	30.78
16	2^{-9}	82.27	18.00
14	2^{-8}	78.42	4.23
12	2^{-7}	73.98	7.81
Performance with Coupled Resonators			
36	2^{-17}	84.57	33.23
18	2^{-9}	84.04	32.84
16	2^{-9}	83.84	31.64
14	2^{-8}	80.10	28.61
12	2^{-7}	68.41	21.81

*Maximum sidelobe ≡ DC gain ÷ maximum gain in stopband.

†Maximum passband ripple ≡ DC gain ÷ maximum peak-to-peak ripple in passband.

were the ripple and sidelobe levels, which were weighed against each other somewhat arbitrarily. When two values of δ yielded equal or nearly equal performance, the larger value was chosen since this would lead to less roundoff noise. The chosen value of δ, which we have called $\delta_{\text{"opt"}}$, and the resulting maximum sidelobe and passband ripple, are shown for each register length. (Note that, according to our definitions, larger numbers in Table II correspond to lower sidelobes and ripple.) For 18 bits, we see that reasonably close to ideal behavior was obtained; but for 16 bits or less, significant degradations occur. The largest degradation is the increased passband ripple. The largest ripples were observed to occur at low frequencies, which can be explained by the fact (see Sec. II-A) that for direct form resonators with low resonant frequencies, the angles of the poles are very sensitive to coefficient quantization. Thus, the deviation of the pole positions increases both with decreasing resonant frequency and with decreasing register length.

The sensitivity to quantization of the low-frequency poles (excluding the real-axis pole) can be reduced if the direct form resonator is replaced by a coupled form. This was tried, replacing (IV-8) by the coupled first-order equations

$$y_k(n) = r\cos\frac{2\pi k}{N}\, y_k(n-1) - r\sin\frac{2\pi k}{N}\, u_k(n-1) + w(n)$$

$$u_k(n) = r\sin\frac{2\pi k}{N}\, y_k(n-1) + r\cos\frac{2\pi k}{N}\, u_k(n-1) \qquad \text{(IV-10)}$$

whose transfer function $Y_k(z)/W(z)$ is the same as in (IV-8) except for a very slight shift in the position of the one z-plane zero. Significant improvements resulted, as we see by reference to Table II. For example, using coupled resonators and 16-bit coefficients, we get better performance than with direct form resonators and 18 bits. Even with 14 bits, the coupled form yields performance reasonably close to the 36-bit standard.

We would expect results similar to these to occur for any low-pass frequency sampling filter, and for such a filter we might well consider using the coupled form, even though execution of (IV-10) requires more multiplications than (IV-8). For a band-pass filter, direct form resonators should probably be used.

A review of Sec. II-B would seem to indicate that, for floating point, the comparison between frequency sampling filters using direct and coupled form resonators would be qualitatively quite similar to the fixed point case. To find out the details of the effects of coefficient quantization on a floating point frequency sampling filter, we would probably have to resort to experiments similar to the above.

B. ROUNDOFF NOISE – FIXED POINT

The analysis of roundoff noise effects in frequency sampling filters is a straightforward application of the techniques of preceding sections. Of particular relevance is Sec. II-C-3, where noise formulas for digital resonators are given. The output noise-to-signal ratio in a frequency sampling filter is strongly dependent on the parameter δ, the distance of the resonator poles from the unit circle. In general, as δ is made smaller, the output noise-to-signal ratio increases, so that we would want to choose the largest possible δ while retaining within some required tolerance the characteristics of the originally designed filter. This issue was considered in Sec. A above, and was taken into account in choosing the values $\delta_{\text{"opt"}}$ in Table II. The effect of roundoff noise will also depend on what realization form is chosen for the digital resonators. Here, we shall consider both the direct and the coupled forms. For frequency sampling filters, we intend to study the roundoff noise problem in detail only for the fixed point case.

Let us refer to the block diagram of Fig. 18, in order to point out the noise sources in a frequency sampling filter. The output of the comb-filter is corrupted by a small amount of white noise due to the multiplication by r^N. The attenuator $1/N$ reduces this comb-filter noise to a negligible level compared with the noise due to roundoff of the multiplication by the attenuator. Thus, at the input to the resonator bank, we have the signal w(n) plus white noise $\epsilon_1(n)$ of variance $\sigma_o^2 = 2^{-2t}/12$ (assuming rounded arithmetic). $\epsilon_1(n)$ propagates through the resonator bank to form one component of the output noise. The remainder of the output noise is the sum of the noises at the resonator outputs, due to roundoffs in the resonators. In computing the output noise variance, the only new calculation needed is the calculation of the effect of $\epsilon_1(n)$, since we can use previous formulas for the noises due to the resonators.

Let us denote by $e_1(n)$ the part of the output noise due to propagation of $\epsilon_1(n)$ through the filter bank, and denote the variance of $e_1(n)$ by $\sigma_{e_1}^2$. Exact calculation of $\sigma_{e_1}^2$ is laborious, since there is overlap between the frequency responses of the filters in the bank. However, since the resonators have very narrow bandwidth, it seems reasonable to neglect this overlap. Then, using high-gain approximations for the variance at each resonator output, and adding variances, we obtain

$$\sigma_{e_1}^2 = \left(\frac{2^{-2t}}{12}\right)\left(\frac{1}{\delta}\right)\left[1/2\, H^2(0) + \sum_{k=1}^{[N/2]} H^2(k)\right] . \tag{IV-11}$$

This formula holds whether direct or coupled form resonators are used. The approximation is valid providing the bandwidth (approximately 2δ) of each resonator is sufficiently small compared with the angular spacing $2\pi/N$ between adjacent resonators. Recall that t denotes the fixed point word length (excluding sign).

Let us denote the total output noise due to roundoffs in the resonators by $e_2(n)$, with variance $\sigma_{e_2}^2$ which will depend on the form chosen for the resonators.

If the direct form is used, then [see (II-30)]

$$\sigma_{e_2}^2 = \left(\frac{2^{-2t}}{12}\right)\left(\frac{1}{\delta}\right)\left[1/2\, H^2(0) + 3\sum_{k=1}^{[N/2]} \frac{H^2(k)}{\sin^2(2\pi k/N)}\right] . \tag{IV-12}$$

[We have assumed that the factors of 2 shown in the resonator transfer functions of Fig. 18 are lumped with the multiplications by H(k).]

Thus, for the case of direct form resonators, the total output noise variance is

$$\sigma_e^2 = \left(\frac{2^{-2t}}{12}\right)\left(\frac{1}{\delta}\right)\left[H^2(0) + \sum_{k=1}^{[N/2]} H^2(k) + 3\sum_{k=1}^{[N/2]} \frac{H^2(k)}{\sin^2(2\pi k/N)}\right] . \tag{IV-13}$$

For the case of a low-pass filter, where the first few H(k) are in the passband, the first few terms in the last sum above will comprise most of σ_e^2, due to the factors $1/\sin^2(2\pi k/N)$ which increase sharply for decreased k.

For the case of coupled form resonators, we have, using (II-43),

$$\sigma_{e_2}^2 = \left(\frac{2^{-2t}}{12}\right)\left(\frac{1}{\delta}\right)\left[1/2\, H^2(0) + 4\sum_{k=1}^{[N/2]} H^2(k)\right] \tag{IV-14}$$

and adding (IV-11) and (IV-14),

$$\sigma_e^2 = \left(\frac{2^{-2t}}{12}\right)\left(\frac{1}{\delta}\right)\left[H^2(0) + 5\sum_{k=1}^{[N/2]} H^2(k)\right] \quad . \tag{IV-15}$$

There are no longer any terms with a $1/\sin^2(2\pi k/N)$ dependence, and thus for low-pass frequency sampling filters there will generally be less total noise variance with coupled form resonators than with direct form resonators.

We have not directly verified experimentally our noise formulas for frequency sampling filters. However, based on the measurements performed by others[20-22] on noise in resonators, we can have reasonable confidence in our models. Actual measurement of noise variance in a frequency sampling filter is a difficult task, since the resonators are very lightly damped, yielding output noise with high correlation from sample-to-sample. Measurement times needed to obtain a consistent estimate of the mean-squared output noise are prohibitive in such a case.

Let us compare (IV-13) and (IV-15) numerically, for our low-pass filter example of the preceding section. [$H(k) = (-1)^k$, $k = 0, 1, \ldots, 31$; $H(32) = 0.7$; $H(33) = -0.225$; $H(34) = 0.01995$; $H(k) = 0$, $k = 35, 36, \ldots, N/2$; $N = 256$.] We assume $\delta = 2^{-9}$, the choice specified in Table II for 18-bit coefficients. For direct form resonators, (IV-13) yields

$$\sigma_e^2 = (345295)2^{-2t} \tag{IV-16a}$$

or, in terms of bits of rms output noise,

$$1/2 \log_2 (2^{2t}\sigma_e^2) = 9.20 \text{ bits} \quad . \tag{IV-16b}$$

Due to the $1/\sin^2(2\pi k/N)$ factors, the first three terms of the last sum in (IV-13) contribute 83.8 percent of the noise variance given in (IV-16a). For the case of coupled form resonators, (IV-15) yields

$$\sigma_e^2 = (6771)2^{-2t} \tag{IV-17a}$$

or in terms of bits,

$$1/2 \log_2 (2^{2t}\sigma_e^2) = 6.36 \text{ bits} \quad . \tag{IV-17b}$$

Thus, for this particular filter, the coupled form yields a decided advantage, about 3 bits lower rms output noise. However, for a filter of a band-pass nature, we would probably wish to use direct form resonators. For a band-pass filter, the noise produced using the direct form would be approximately the same as with the coupled form, and the direct form would have the advantage of increased computational speed.

In the above comparison, we have not mentioned the overflow constraint. However, for the case of a frequency sampling filter, the output signal power that can be attained while satisfying the overflow constraint is essentially the same for direct and coupled form resonators, so that the comparison above is fair.

To prevent overflow in a frequency sampling filter, we require that (1) the comb-filter does not overflow, and (2) that none of the resonators overflow. If these requirements are satisfied, then any overflow caused by summing the outputs of the resonators can be prevented by attenuators at the resonator outputs. Since these final attenuators act on both signal and noise, they would not affect the output noise-to-signal ratio.

For the comb-filter, we have $\sum_{n=0}^{\infty} |h(n)| \leqslant 2$, so that overflow can be prevented by restricting $|x(n)| < 1/2$. To prevent overflow in the resonators, we insert an attenuator between the comb-filter and resonator bank, as indicated in Fig. 18. Without the attenuator, the kth comb-filter-resonator combination has z-transform

$$H_{cr}(z) = \frac{(1 - r^N z^{-N})\,[1 - r\cos(2\pi k/N)\,z^{-1}]}{1 - 2r\cos(2\pi k/N)\,z^{-1} + r^2 z^{-2}}$$

and impulse response

$$h_{cr}(n) = \begin{cases} r^n \cos\left(\frac{2\pi kn}{N}\right) \quad , & n = 0, 1, \ldots, N-1 \\ 0 \quad , & \text{otherwise} \quad . \end{cases} \tag{IV-18}$$

It can easily be verified that

$$\sum_{n=0}^{\infty} |h_{cr}(n)| < N \tag{IV-19}$$

so that with $|x(n)| < 1/2$, an attenuator $2/N$ will prevent overflow in the resonator. Another way of formulating the condition for no overflow in a frequency sampling filter, is to revise Fig. 18 so that an attenuator $1/2$ precedes the comb-filter, and an attenuator $2/N$ follows the comb-filter. Then, our restriction on the input signal is simply $|x(n)| < 1$.

Given these conditions, we can obtain expressions for output noise-to-signal ratio in our filter. As usual, the noise-to-signal ratio depends on the type of input signal used. We consider a white input, with first-order probability density function uniform in $(-1, 1)$, so that $\sigma_x^2 = 1/3$. The output signal variance can be expressed as

$$\sigma_y^2 = \frac{H^2(0) + 2\sum_{k=1}^{[N/2]} H^2(k)}{N}\,\sigma_x^2 \quad . \tag{IV-20}$$

For the filter example we have been considering,

$$\sigma_y^2 \approx \frac{64}{256}\,\sigma_x^2 = \frac{1}{12} \quad . \tag{IV-21}$$

Since the filter is fairly wideband, a sizable portion of the input signal power passes through the filter. These formulas are valid for both direct and coupled form resonators, and can be used in conjunction with the above results for output noise variance to yield formulas for output noise-to-signal ratio.

By using (IV-16) and (IV-21), the rms output noise-to-signal ratio, for direct form resonators, is (expressed in units of bits)

$$1/2\,\log_2\left[2^{2t}\,\frac{\sigma_e^2}{\sigma_y^2}\right] = 10.99 \text{ bits} \quad . \tag{IV-22}$$

By using (IV-17) and (IV-21), the corresponding result for coupled form resonators is

$$1/2\,\log_2\left[2^{2t}\,\frac{\sigma_e^2}{\sigma_y^2}\right] = 8.15 \text{ bits} \quad . \tag{IV-23}$$

C. SUMMARY OF QUANTIZATION EFFECTS IN FREQUENCY SAMPLING FILTERS

A frequency sampling filter is a realization of a finite duration impulse response filter by means of a comb-filter and a bank of resonators. This realization technique is quite flexible, since any finite duration impulse response filter can be synthesized in this way. The design of such a filter can be accomplished by specifying samples of the filter's frequency response; these samples correspond to constant multipliers to be applied to the outputs of the resonators. For a band- or low-pass filter, where the out-of-band frequency samples are specified to be zero, the number of resonators needed may be modest, and thus the amount of computation time needed to compute each output point may compare favorably with the time needed in a direct convolution, or in an FFT realization. The frequency sampling realization can also be used to synthesize a bank of band-pass filters. One comb-filter can be shared by all the resonators in the bank.

Quantization problems in a frequency sampling filter are quite severe, due to the need for pole-zero cancellations near the unit circle. Because of coefficient quantization, the pole-zero cancellations will be imperfect. This can lead, for example, to large passband ripples in the filter's frequency response. For a particular low-pass filter example, experiments have indicated that, with direct form resonators, 16-bit coefficients (fixed point) are needed to keep passband ripple tolerably low. If the resonators are realized using the coupled form, so that the sensitivity to coefficiency errors of the resonant frequency of the low-resonant-frequency poles is reduced, then 14 bits seem sufficient.

Since high-gain resonators are required in the frequency sampling realization, the effects of roundoff noise and the overflow constraint are also quite severe. Thus, quite accurate arithmetic would be needed to maintain a suitably low noise-to-signal ratio. For the fixed point case, a noise-to-signal ratio analysis was carried out, and numerical results were obtained for our low-pass filter example. Expressing the results in decibels, we have from (IV-22),

$$\left.\frac{N}{S}\right)_{dB} = 10 \log_{10} \sigma_e^2/\sigma_y^2 = 66 - 6t \qquad \text{(IV-24)}$$

if direct form resonators are used. To maintain $N/S \leqslant -60\,dB$, we would need a word length $t = 21$ bits. For coupled form resonators, the situation is somewhat improved, as from (IV-23) we have

$$\left.\frac{N}{S}\right)_{dB} = 48.9 - 6t \qquad \text{(IV-25)}$$

so that to maintain a -60-dB N/S, $t = 18$ bits would approximately suffice.

We should comment that the advantage of coupled form resonators, with respect to both coefficient sensitivity and noise-to-signal ratio, holds only for low-pass filters. For a band-pass filter, the effects of quantization for direct and coupled form resonators are quite similar, and we would probably use the direct form to obtain greater computational speed.

V. FFT FILTERS

In this section, we consider quantization effects in the class of digital filters which have an impulse response of finite duration and are implemented by means of convolution sums performed using the discrete Fourier transform (DFT). The output samples of such a filter are obtained from the results of N-point circular convolutions of the filter impulse response (kernel) with sections of the input. These circular convolutions are obtained by computing the DFT of the input section, multiplying by the DFT of the impulse response, and inverse transforming the results. Stockham[2,7] has discussed procedures for utilizing the results of these circular convolutions to perform linear convolutions, methods for piecing together sections of output, rationales for choosing the transform length N, and speed advantages over direct convolution to be gained by using the FFT to implement the required DFT's. We shall not discuss the details of these issues, but concern ourselves with the effects of quantization on the basic computation, involving an FFT of input section, multiplication by DFT of kernel, and inverse FFT. The additional error caused by piecing sections together will generally be negligible compared with the errors due to the FFT-multiply-IFFT (inverse fast Fourier transform). We shall restrict attention to radix 2 FFT algorithms.

Any filter with finite duration impulse response can be implemented via the route of the FFT, as well as by the frequency sampling technique given in Sec. IV. In working out numerical results in this section, we shall use the same prototype filter used in Sec. IV. This filter has an impulse response of duration 256 samples, and we shall assume in its FFT implementation that 512-point circular convolutions are used, so that an input section might consist of 256 input values augmented by 256 zeros.

We shall draw heavily in this section on the results presented in Sec. III on quantization in the FFT. A few of the issues, such as the arranging of scaling to prevent overflow, change somewhat for the case of an FFT filter.

Most of our attention here will be confined to the fixed point case. To detail the way in which matters change for a block floating or floating point realization would be to repeat a great deal of our preceding discussion on the FFT.

A. COEFFICIENT QUANTIZATION

Because of coefficient quantization, the impulse response and frequency response of an FFT filter will deviate from the specified design. Analytical treatment of this problem presents many difficulties. The analysis in Sec. III-A, which dealt with mean-squared error due to coefficient quantization in an FFT, is not particularly useful for this problem. We would be more interested in how quantities such as the maximum passband ripple and maximum sidelobe depend on coefficient quantization, than in a mean-squared error criterion. We have not been able to analyze theoretically these effects. One of the problems is that the impulse response of an FFT filter (with quantized coefficients) is not well defined, for the response of an FFT filter will, in general, change (not merely by a time translation) if the position of the input pulse is shifted. When the impulse is shifted, different coefficients, with different quantization errors, will be used in computing the FFT.

For these reasons, we have used an empirical approach in estimating the effects of coefficient quantization on an FFT filter. The low-pass filter design specified in Sec. IV-A was used as a prototype. The 512-point FFT of an impulse input was computed, using coefficients rounded to a chosen number of bits M (18 or less) and 36-bit arithmetic to minimize the effect of roundoff noise. The resulting sequence was multiplied by a 512-point DFT of the filter impulse response. In obtaining this filter function, the 256-point filter impulse response was obtained by inverse transforming the frequency samples specified in Sec. IV-A; 256 zeros were appended, and a 512-point DFT was computed (36-bit arithmetic and coefficients were used in these computations, but the resulting 512-point sequence was rounded to M bits). The product sequence (DFT of input section times DFT of filter) was then inverse transformed using M-bit coefficients and 36-bit arithmetic. This yielded a filter impulse response, corrupted due to coefficient quantization. To obtain a frequency response, this impulse response was lengthened to 2048 points by appending zeros, and a 2048-point $|\mathrm{DFT}|^2$ was computed using 36-bit arithmetic and coefficients. All the above arithmetic was fixed point.

As a control, M was initially set to 36 bits, and the "ideal" frequency response specified in Fig. 19 was obtained. As M was reduced, measurements showed the maximum sidelobe and passband ripple to increase. Scope pictures showed that the degradation in frequency response appeared in the form of noise distributed fairly uniformly over the entire frequency axis. This was in contrast to the results for frequency sampling filters, where spurious ripples in the frequency response clustered near the frequencies of a few uncanceled resonator poles.

The experimental results are summarized in Table III, which show maximum sidelobe and passband ripple as a function of coefficient register length (including sign). Recall that larger numbers in the table correspond to lower sidelobes and ripple. For each register length, several measurements of sidelobe and ripple were made, corresponding to different positions (in time) of the input impulse. Only slight variations of sidelobe and ripple with impulse position were observed, and the results shown in the table represent an average over the measurements made.

We can compare Table III with the corresponding results given in Table II for the frequency sampling realization. With respect to passband ripple, the FFT filter generally gives better performance for equal register length. For the FFT realization, we do not have the problem of uncanceled poles in the passband, and the register length must be reduced quite drastically before the uniform noise in the filter's frequency response becomes comparable in level to the passband ripples which were already present in the originally designed filter. However, the frequency sampling realization generally has somewhat better sidelobe rejection, for equal register length. A partial explanation for this is that in a frequency sampling realization with reduced register length, the zeros of the comb-filter can still be kept fairly close to the unit circle, maintaining the stopband attenuation. In the FFT filter, the noise level in the stopband of the frequency response increases fairly uniformly with decreased register length.

B. A BOUND ON THE OUTPUT OF A CIRCULAR CONVOLUTION†

Here, we derive a bound on the output of an N-point circular convolution. This bound is useful in arranging scaling of the array to prevent overflow in an FFT filter. The bound is derived for general filter characteristics, and thus would be useful, for example, in setting up a program or hardware to implement an FFT filter with arbitrary frequency response characteristics. For specific filter characteristics, bounds can be derived which are lower than the general bound. In addition to the general bound, we shall evaluate a specific bound for the low-pass filter example of the preceding sections. In deriving the general bound, the style of this section will become somewhat more mathematical than that of the remainder of the report.

† Much of this work has been reported on previously by Oppenheim and the author.[30]

TABLE III

PERFORMANCE OF FFT FILTER WITH QUANTIZED COEFFICIENTS

Register Length	Maximum Sidelobe (dB)	Maximum Passband Ripple (dB)
36	84.57	33.23
18	83.45	33.23
16	80.00	33.22
14	79.30	33.22
12	66.44	33.21
10	55.34	33.20
8	42.31	29.46

1. Problem Statement

According to the above discussion, we would like to determine an upper bound on the maximum modulus of an output value that can result from an N-point circular convolution. With $\{x(n)\}$ denoting the input sequence, $\{h(n)\}$ denoting the kernel, and $\{y(n)\}$ denoting the output sequence, we have

$$y(n) = \sum_{k=0}^{N-1} x(k)\, h[(n-k) \bmod N] \qquad n = 0, 1, \ldots, N-1 \tag{V-1}$$

where it is understood that, in general, each of the three sequences may be complex. The circular convolution is accomplished by forming the product

$$Y(k) = H(k)\, X(k) \tag{V-2}$$

where

$$X(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n)\, W^{nk} \qquad k = 0, 1, \ldots, N-1 \tag{V-3}$$

$$Y(k) = \frac{1}{N} \sum_{n=0}^{N-1} y(n)\, W^{nk} \qquad k = 0, 1, \ldots, N-1 \tag{V-4}$$

and

$$H(k) = \sum_{n=0}^{N-1} h(n)\, W^{nk} \qquad k = 0, 1, \ldots, N-1 \tag{V-5}$$

with W defined as $W = e^{-j2\pi/N}$

We imagine the computation to be carried out on fixed point fractions. Thus, we bound the input values so that

$$|x(n)| \leqslant 1 \quad . \tag{V-6}$$

By virtue of (V-3), we are then assured that

$$|X(k)| \leqslant 1$$

so that the values of X(k) do not overflow in the fixed point work. If we are dealing with a radix 2 FFT algorithm, the factor 1/N in (V-3) can be incorporated as scalings by 1/2 at each array. As discussed in Sec. III-C, these scalings will guarantee no overflows in the intermediate arrays and will keep thc signal level as high as possible as we proceed through the computation.

In the typical case, the sequence h(n) is known and, consequently, so is the scquence H(k). Therefore, it is not necessary to continually evaluate (V-5); that is, the sequence H(k) is computed, normalized, and stored in advance. Thus, it is reasonable to only apply a normalization to H(k) and not to h(n), so that we require that

$$|H(k)| \leqslant 1 \quad . \tag{V-7}$$

A normalization of the transform of the kernel so that the maximum modulus is unity allows maximum energy transfer through the filter, consistent with the requirement that Y(k) does not overflow the register length.

Our objective is to obtain an upper bound on $|y(n)|$ for all sequences $\{x(n)\}$ and $\{H(k)\}$ consistent with (V-6) and (V-7). This bound will specify, for example, the scaling factor to be applied in computing the inverse of (V-4) to guarantee that no value of y(n) overflows the fixed point word. The results which we will obtain are that:

(a) With the above constraints, the result of the N-point circular convolution of (V-1) is bounded by $|y(n)| \leqslant \sqrt{N}$.

(b) In the general case, where $\{x(n)\}$ and $\{h(n)\}$ are allowed to be complex, the bound in A is a least upper bound. This will be shown by demonstrating a sequence that can achieve the bound.

(c) If we restrict $\{x(n)\}$ and/or $\{h(n)\}$ to be real-valued, the bound of A is no longer a least upper bound for every N. However, the least upper bound $\beta(N)$ is itself bounded by $\sqrt{N}/2 \leqslant \beta(N) \leqslant \sqrt{N}$.

2. Derivation of Results

Proof of (a)

Parseval's relation requires that

$$\sum_{n=0}^{N-1} |y(n)|^2 = N \sum_{k=0}^{N-1} |Y(k)|^2 \tag{V-8}$$

and

$$\sum_{n=0}^{N-1} |x(n)|^2 = N \sum_{k=0}^{N-1} |X(k)|^2 \quad . \tag{V-9}$$

By substituting (V-2) into (V-8), and using (V-7),

$$\sum_{n=0}^{N-1} |y(n)|^2 \leqslant N \sum_{k=0}^{N-1} |X(k)|^2 \tag{V-10}$$

or, using (V-9),

$$\sum_{n=0}^{N-1} |y(n)|^2 \leqslant \sum_{n=0}^{N-1} |x(n)|^2 \tag{V-11}$$

with equality, if and only if $|H(k)| = 1$. However, (V-6) requires that

$$\sum_{n=0}^{N-1} |x(n)|^2 \leqslant N \tag{V-12}$$

with equality, if and only if $|x(n)| = 1$. By combining (V-11) and (V-12),

$$\sum_{n=0}^{N-1} |y(n)|^2 \leqslant N \quad . \tag{V-13}$$

But,

$$|y(n)|^2 \leqslant \sum_{n=0}^{N-1} |y(n)|^2 \tag{V-14}$$

and, therefore,

$$|y(n)| \leqslant \sqrt{N} \quad . \tag{V-15}$$

Proof of (b)

To show that $\sqrt{N}$ is a least upper bound on $|y(n)|$, we review the conditions for equality in the inequalities used above. We observe that for equality to be satisfied in (V-15), it must be satisfied in (V-11), (V-12), and (V-14), requiring that

(1) $|H(k)| = 1$

(2) $|x(n)| = 1$

(3) Any output sequence y(n) which has a point whose modulus is equal to $\sqrt{N}$ can contain only one nonzero point.

The third requirement can be rephrased as a requirement on the input sequence, and on the sequence H(k). Specifically, if the output sequence contains only one nonzero point, then Y(k) for this sequence must be of the form

$$Y(k) = AW^{n_o k} = |A| \exp[-j(\tfrac{2\pi}{N} n_o k + \rho)]$$

where ρ is a real constant and n_o is an integer so that, from (V-2),

$$H(k)\,X(k) = |A| \exp[-j(\tfrac{2\pi}{N} n_o k + \rho)] \quad . \tag{V-16}$$

We can express H(k) and X(k) as

$$H(k) = e^{j\eta_k} \tag{V-17a}$$

and

$$X(k) = |X(k)| e^{j\theta_k} \tag{V-17b}$$

where we have used the fact that $|H(k)| = 1$. For (V-16) to bc satisfied, then

$$|X(k)| = |A|$$

and

$$\eta_k = -\theta_k - \frac{2\pi}{N} n_o k - \rho \quad . \tag{V-18}$$

Therefore, requirement (3) can be replaced by the statement that:

(3)' $|X(k)|$ = constant and the phase of H(k) be chosen to satisfy (V-18).

As an additional observation, we note that for <u>any</u> input sequence $\{x(n)\}$,

$$|y(n)| \leqslant \sum_{k=0}^{N-1} |H(k)|\ |X(k)|$$

with equality for some value of n, if and only if $|H(k)| = 1$ and the phase of H(k) is chosen on the basis of (V-18). Therefore, for any x(n) the output modulus is maximized when H(k) is chosen in this manner. However, this maximum value will only equal $\sqrt{N}$ if, in addition, $|x(n)| = 1$ and $|X(k)|$ = constant.

For N even, a sequence having the property that $|x(n)| = 1$ and $|X(k)|$ = constant is

$$x(n) = e^{-j(\pi n^2/N)} = W^{n^2/2} \quad . \tag{V-19}$$

For N odd, a sequence with $|x(n)| = 1$ and $|X(k)|$ = constant is†

$$x(n) = e^{-j(2\pi n^2/N)} = W^{n^2} \quad . \tag{V-20}$$

By using one of these sequences as the input, and choosing $H(k) = e^{j\eta_k}$, with η_k given by (V-18), equality in (V-15) can be achieved for any N. Thus, the bound given in statement (a) is a least upper bound.

Proof of (c)

Proof of (c) is omitted here. See Oppenheim and Weinstein.[30]

3. Application of General Bounds

In a fixed point FFT filter, we wish to incorporate scaling in the computation in such a way that we are guaranteed never to overflow. When we use a power of two FFT algorithm, overflows in the computation of X(k) will be prevented by including a scaling of 1/2 at each stage, since the maximum modulus of an array in the computation is nondecreasing and increases by

† The sequences (V-19) and (V-20) were suggested by C.M. Rader of Lincoln Laboratory. For proof of their stated properties, see Oppenheim and Weinstein.[30]

at most a factor of two as we pass from stage to stage (see Sec. III-A). With this scaling, the bound just derived guarantees that scaling is not required in more than half the arrays of the inverse FFT computation. Therefore, including a scaling of 1/2 in the first half of the stages in the inverse FFT will guarantee that there are no overflows in the remainder of the computation. For general filter characteristics, no less scaling will suffice to guarantee against overflow, and no more scaling is necessary. The fact (c) in Sec. 2 above, that $\beta \geqslant N/2$, indicates that if we restrict ourselves to real input data or if h(n) is real, at most one rescaling could be eliminated.

The bound just obtained is also useful if the FFT's are carried out using a block floating point strategy. In this case, arrays are scaled by a factor of 1/2 only when overflows occur, but then a final rescaling must be carried out after each section is processed so that it is compatible with the results from previous sections. For general input filter characteristics, the final rescaling can be chosen based on the bounds given here, to insure that the output will not exceed the available register length.

4. Bounds for Specific Filter Characteristics

The bounds derived and method of scaling mentioned above, apply to the general case. Except for the normalization of (V-7), they do not depend on the filter characteristics. This is useful when we wish to fix the scaling strategy without reference to any particular filter. For specific filter characteristics, the bound can be reduced. Specifically, it can be verified from (V-1) and (V-6) that, in terms of h(n),

$$|y(n)| \leqslant \sum_{\ell=0}^{M-1} |h(\ell)| \tag{V-21}$$

where M denotes the length of the impulse response. This is a least upper bound since a sequence $\{x(n)\}$ can be selected which will result in this value in the output. This will be significantly lower than the bound represented in (V-15) if, for example, the filter is very narrow band, or if the kernel has many points with zero value.

The bound (V-21) has been evaluated for the low-pass filter example which was specified in Sec. IV-A. For this filter, we have M = 256, and the impulse response is

$$h(n) = \frac{2}{256}\left\{\frac{1}{2} + \sum_{k=0}^{31} (-1)^k \cos\frac{2\pi kn}{256} + 0.7\cos\frac{2\pi(32)n}{256} \right.$$
$$\left. - 0.225\cos\frac{2\pi(33)n}{256} + 0.01995\cos\frac{2\pi(33)n}{256}\right\} \quad . \tag{V-22}$$

We found that

$$\sum_{n=0}^{255} |h(n)| = 3.12 \quad . \tag{V-23}$$

If, as suggested previously, the filter is implemented using 512-point FFT's, then a scaling by 1/2 at just the first two stages of the inverse transform will suffice to prevent overflow. If the general bound were used instead, scaling at the first five arrays would be applied. Thus, use of the specific bound rather than the general bound permits the signal level to be kept significantly higher, implying a corresponding improvement in output noise-to-signal ratio.

C. ROUNDOFF NOISE – FIXED POINT

Using the results of our roundoff noise analysis for fixed point FFT, we can obtain expressions for output noise variance and output noise-to-signal ratio, for a fixed point FFT filter. Let us examine the basic steps in the filtering computation, tracing the buildup of noise variance as we proceed. As in the preceding sections, we shall focus on a prototype filter with 256-point impulse response. We assume rounded arithmetic.

First, we use the FFT to compute the DFT of a section of input. In the implementation of our prototype filter, which has a 256-point impulse response, we might compute a 512-point FFT, where the input consists of 256 input samples and 256 zeros. Actually, we could handle 512 real input samples at once, by placing sections of 256 real samples in both the real and imaginary parts of the input to the FFT. To guarantee against overflow, we need a scaling of 1/2 at each of the 9 stages in the FFT, yielding an overall attenuation of 1/512. Our requirement on the input sequence is $|x(n)| < 1$. The DFT we compute will be corrupted by noise, which we call $E_1(k)$. This noise has variance [see (III-24) and (III-26)]

$$\sigma^2_{E_1} = \mathcal{E}|E_1(k)|^2 = (5/3)2^{-2t} \quad . \tag{V-24}$$

This noise variance is small, because most of the roundoff noise has been shifted off by the attenuations. However, due to the scalings, the mean-squared signal has decreased by a factor of 1/512, so that the noise-to-signal ratio has increased significantly in passing through the FFT.

Next, we multiply this computed transform by a sequence $H(k)$ representing the DFT of a 512-point sequence $\tilde{h}(n)$, where $\tilde{h}(n)$ consists of the filter impulse response and 256 zeros. This multiplication introduces new roundoff noise of variance $2^{-2t}/3$.[†] If we assume $|H(k)| < 1$, the mean-square of the noise which was already present due to the first FFT gets reduced by a factor

$$\frac{1}{512} \sum_{k=0}^{511} |H(k)|^2 \equiv B \tag{V-25}$$

representing a ratio of the filter bandwidth to the sampling frequency. Thus, after the multiplication, the mean-square of the total noise $E_2(k)$ is approximately

$$\sigma^2_{E_2} = \frac{2^{-2t}}{3} + B(5/3)2^{-2t} \quad . \tag{V-26}$$

The noise is not white, but contains a component whose spectrum has been shaped by the filter.

For our low-pass filter example, $B \approx 1/4$, so that

$$\sigma^2_{E_2} \approx (3/4)2^{-2t} \quad . \tag{V-27}$$

Note that $\sigma^2_{E_2}$ is slightly less than $\sigma^2_{E_1}$, and represents only about a bit of noise. If the signal spectrum is flat, the mean-squared signal will also be reduced somewhat due to the multiplication by $H(k)$.

† H(k) is, in general, complex, so that four real multiplies are needed. If certain points on H(k) are real or zero, less noise variance will be introduced.

Now we compute an inverse transform to obtain a section of output. The noise variance at the output of this transform depends on how many scalings are necessary in the inverse FFT. Let us consider our low-pass filter example, and assume that the specific bound of Sec. V-B-4 is used, so that only two scalings (at the first two arrays) are necessary in the inverse transform. Then, in propagating through the IFFT, the mean-square of the noise $E_2(k)$ increases by a factor of 512/16. The 512 represents the gain of the inverse DFT, and the 1/16 is due to scalings. The variance of the additional output $E_3(k)$ noise caused by roundoff in the IFFT can be estimated easily via the method of Sec. III-B. The result is

$$\sigma_{E_3}^2 = \sigma_{B'}^2 \left(\frac{512}{8} + \frac{512}{4}\right) + \sigma_B^2 \left(\frac{512}{8} + \frac{512}{4} + \ldots + 1\right) = (202.3)2^{-2t} \quad . \tag{V-28}$$

The total mean-squared output noise is

$$\sigma_E^2 = (512/16)\sigma_{E_2}^2 + \sigma_{E_3}^2 \approx (226)2^{-2t} \quad . \tag{V-29}$$

By representing this result in units of bits of rms output noise,

$$\frac{1}{2} \log_2 (2^{2t}\sigma_E^2) = 3.91 \text{ bits} \quad . \tag{V-30}$$

We can estimate the mean-squared output signal, if we assume specific statistics for the input signal x(n). Let us assume x(n) is white, with real and imaginary parts uniformly distributed in $(-1, 1)$. Its mean-square is then

$$\sigma_x^2 \equiv \mathcal{E}|x(n)|^2 = 2/3 \quad .$$

This variance goes through an attenuation of 1/512 in the first FFT (due to the scalings at each array), an attenuation of $B = 1/4$ due to the multiplication by H(k), and a gain of 512/16 in the inverse transform. Thus, the mean-squared output signal is

$$\sigma_y^2 = (1/512)B\,(512/16)\sigma_x^2 = \frac{1}{96} \quad . \tag{V-31}$$

The output noise-to-signal ratio is

$$\sigma_E^2/\sigma_y^2 = (21696)2^{-2t} \tag{V-32}$$

or in units of bits,

$$\frac{1}{2} \log_2 [2^{2t}(\sigma_E^2/\sigma_y^2)] = 7.20 \text{ bits} \quad . \tag{V-33}$$

Comparing this result with the corresponding results (IV-22) and (IV-23) for the frequency sampling realization, we see that, for this example, the FFT filter provides somewhat better noise-to-signal ratio. The advantage is 3.79 bits if direct form resonators are used in the frequency sampling filter, but only 0.95 bit if coupled form resonators are used.

D. COMPARISON OF FFT AND FREQUENCY SAMPLING FILTER REALIZATION

One of the stated objectives of the last two sections was to compare, on the basis of differing quantization effects, FFT and frequency sampling realizations of the same digital filter. The techniques for constructing such a comparison have been demonstrated, and have been applied to a specific design example for which (we considered the case of fixed point arithmetic) the FFT realization yielded a slightly lower output noise-to-signal ratio. The effects of coefficient quantization were qualitatively somewhat different for the two realizations. For equal register length coefficients, we might judge either realization to yield better performance, depending on our criterion. Considering both coefficient quantization and noise, we would slightly prefer the FFT realization.

A more general comparison of the FFT and frequency sampling realizations, on the basis of differing quantization effects, seems to us to be both difficult and unprofitable. Two of the difficulties are that first, a different comparison needs to be constructed for each specific filter design, and second, it is not clear, in general, what the criterion for comparison should be. This more general comparison would also be somewhat unprofitable, since there are many factors other than quantization effects which must be considered in choosing between alternate filter designs. For example, a filter (of relatively long impulse response) in which all or most of the frequency samples H(k) were nonzero would almost certainly be realized by the FFT technique, due to an advantage in computational speed over the frequency sampling method. The FFT realization is a versatile method for the implementation of finite duration impulse response filters, while the frequency sampling method, although general, is practical in only the special cases where all but a few of the frequency samples are zero. For example, the frequency sampling technique would be quite appropriate for the realization of a bank of fairly narrow band-pass filters. As a general synthesis procedure for finite duration impulse response filters, we would almost certainly prefer the FFT method.

Apart from setting forth a comparison, a perhaps more important objective of Secs. IV and V was to demonstrate how the techniques and results developed in studying quantization effects in simple recursive filters and the FFT, could be applied to the analysis of more complicated systems. A point we wished to make was that, given a good understanding of quantization effects in the basic algorithms (simple recursive filters and the FFT), the analysis of more complicated systems can be accomplished straightforwardly.

VI. CONCLUSIONS AND PROBLEMS FOR FURTHER STUDY

In Sec. A below, we now summarize the report and review the important results. Then, in Sec. B, we mention some problems which have not been resolved here and which seem worthy of further study.

A. SUMMARY AND REVIEW OF RESULTS

We began Sec. I by outlining the problem area of quantization effects in digital filters; this outline is presented in brief form at the end of Sec. I-A. After reviewing previous work relevant to the problem area (Sec. I-B), we devoted Sec. I-C to a study of roundoff error models, both fixed and floating point. The basic statistical roundoff error models, which are used throughout the report, were set down and a few experiments testing some of the assumptions of the models were reported. In general, the experimental results were in accord with the assumptions of the roundoff error models.

Section II concentrated on quantization effects in simple (first- and second-order) recursive filters. The analysis of coefficient quantization was reviewed for the fixed point case, and carried over to the floating point case. We found that when the range of coefficient magnitudes was not too large, floating point provided little advantage over fixed point with respect to the coefficient quantization problem. Thus, for many practical filters, we might well consider using fixed point coefficients (thus saving the bits needed for the coefficient exponents) even if floating point arithmetic is to be employed.

The effects of roundoff noise and the overflow constraint were considered jointly to derive output noise-to-signal ratios for fixed, floating, and block floating point filter realizations. For the floating point case, experiments verifying the theory were presented; for the fixed point case, the use of a clamp to prevent overflow was investigated.

For both fixed and floating point arithmetic, we compared (on the basis of differing output noise-to-signal ratios) direct, canonic, and coupled form digital resonator realizations. We found that for a filter with low resonant frequency, the coupled form yields a lower noise-to-signal ratio than either the direct or canonic forms, which yield about equal noise-to-signal ratios.

For the case of a white input signal, we compared noise-to-signal ratios for fixed, floating, and block floating point realizations of the same filter. We found that, as the gain of the filter increases, the noise-to-signal ratio for fixed point increases at a greater rate than for floating or block floating point. This comparison, however, is sensitive to the assumption of white input signal, since the fixed point noise-to-signal ratio depends quite strongly on the type of input signal. However, the methods used for deriving the noise-to-signal ratios can be applied for any type of input signal.

In Sec. III, we studied quantization effects for the other basic algorithm of digital filtering, the FFT. The results are summarized in detail in Sec. III-E, so only brief comments will be given here. For fixed point, we found that the effect of coefficient quantization was to produce a mean-squared output error-to-signal ratio proportional to $\nu = \log_2 N$, whereas roundoff noise caused a noise-to-signal ratio proportional to N. Thus, for fixed point, it would be reasonable to maintain fewer bits in the coefficient accuracy than in the accuracy of the arithmetic. We found that for signals with a fairly flat spectrum, a block floating point strategy could yield significant improvements in output noise-to-signal ratio, over a fixed point strategy. The mean-squared output noise-to-signal ratio for a floating point FFT of a white signal was derived (using a statistical roundoff error model), and found to be proportional to $\nu = \log_2 N$. This result disagreed with the experimental results of previous workers, who measured noise-to-signal ratios proportional to ν^2. We found, however, that if "randomized" rounding (see Sec. III-D-3) were used, experimental results proportional to ν were obtained. With "nonrandomized" rounding, or truncation, our measured noise-to-signal ratios seemed to be proportional to ν^2, in agreement with the results of previous workers.

Section IV was devoted to quantization effects in frequency sampling filters, and most of our attention was restricted to the fixed point case. The coefficient accuracy problem was studied in some detail. For a particular filter example, we measured sidelobe level and passband ripple, as a function of the number of bits used for the coefficients. We showed that for a low-pass frequency sampling filter, the use of coupled form rather than direct form resonators significantly reduced the sensitivity to coefficient quantization. We applied the results of Sec. II to derive output noise-to-signal ratio for a fixed point frequency sampling filter. We found that, for a low-pass filter, a lower output noise-to-signal ratio was obtained with coupled form resonators than with direct form resonators.

In Sec. V, we analyzed quantization effects in FFT filters. The coefficient quantization problem was investigated empirically. For the same low-pass filter example studied in Sec. IV, we measured sidelobe level and passband ripple as a function of the number of bits retained in the coefficients. Compared with the frequency sampling realization, the FFT filter yields lower passband ripple but somewhat higher sidelobe level, for equal word length coefficients.

One of the important questions in setting up a fixed point FFT filter realization is how to arrange the scaling of the array to prevent overflow, yet minimize output noise-to-signal ratio. We derived (Sec. V-B) a general bound on the output of a circular convolution, which aids in arranging the scaling, for arbitrary filter characteristics. We also gave a bound which is applicable for specific filter characteristics, and is usually lower than the general bound. Using this specific bound, we showed how scaling would be arranged in the low-pass filter example mentioned above.

With this scaling scheme set, we applied the techniques of Sec. III to derive output noise-to-signal ratio for a fixed point, FFT realization of our low-pass filter example. The derived noise-to-signal ratio was somewhat lower than for a frequency sampling realization of the same filter.

B. PROBLEMS FOR FURTHER STUDY

In conclusion, let us mention some problems which have been studied here slightly or not at all, and which we feel are worthy of further research.

Most of our work dealt with situations where the quantization was sufficiently fine so that linear noise analysis was applicable. It would be interesting to study systems where the quantization is very coarse. Consider, for example, a second-order recursive filter with 5-bit arithmetic. Such a system probably cannot be satisfactorily analyzed as a linear system with additive quantization noise. The system is highly nonlinear, and would exhibit such effects as complicated limit cycles. We would probably want to use clamping in such a system, and this would complicate the analysis even further. However, the system has the advantage that hardware requirements are low, and computational speed could be very high. A study of such highly quantized systems, finding practical applications for them and some theory to describe their operation, would be of much interest. This type of study need not be restricted to recursive filters, but could include the FFT as well. For example, the effects of very coarse coefficient quantization in an FFT might be investigated.

Another problem area is the question of quantization effects in a very long (say 10^6-point) or multidimensional FFT. The theory we developed in Sec. III can be used, to some extent, to analyze this problem. But to know for certain whether our results can be extrapolated straightforwardly for very long transforms, we would need to perform experiments. Just programming a 10^6-point FFT could be a large project, since most computer memories could not accommodate all the data, and efficient use of secondary storage (drums, disks, tapes) would have to be arranged.

In an FFT filter, it is often advantageous to do simultaneous filtering of two real signals, using one as the real part and the other as the imaginary part of the input to the filter. Because of quantization, there will be crosstalk between these two signals, that is, each of the two output signals will be affected by both input signals. This problem deserves some investigation.

Finally, all our theoretical quantization noise analysis has treated the case of rounding, rather than truncation. Some theoretical work for the case of truncation would be useful. For example, could it be explained theoretically why, in a floating point FFT with truncation, the experimental noise-to-signal ratios tend to be proportional to $(\log_2 N)^2$ rather than to $\log_2 N$ as for rounding? This problem involves modifying the roundoff error model to account for the signal-noise correlation caused by truncation; the analysis would probably be fairly tedious.

REFERENCES

1. J. F. Kaiser, "Digital Filters," in System Analysis by Digital Computer, by F. Kuo and J. F. Kaiser (McGraw-Hill, New York, 1959), Chapter 7, pp. 218-285.
2. B. Gold and C. M. Rader, Digital Processing of Signals (McGraw-Hill, New York, 1969).
3. R. M. Golden and J. F. Kaiser, "Design of Wideband Sampled-Data Filters," Bell System Tech. J. 43, 1533-1546 (1964).
4. C. M. Rader and B. Gold, "Digital Filter Design Techniques in the Frequency Domain," Proc. IEEE 55, 149-171 (1967), DDC AD-651530.
5. J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," Math. Comput. 19, 297-301 (1965).
6. W. T. Cochran, et al., "What Is the Fast Fourier Transform?," Proc. IEEE 55, 1664-1674 (1967).
7. T. G. Stockham, Jr., "High Speed Convolution and Correlation," Proceedings Spring Joint Computer Conference, Vol. 28 (1966), pp. 229-233.
8. J. F. Kaiser, "Some Practical Considerations in the Realization of Linear Digital Filters," Proceedings Third Allerton Conference on Circuit and System Theory, Monticello, Illinois, October 1965, pp. 621-633.
9. C. M. Rader and B. Gold, "Effects of Parameter Quantization on the Poles of a Digital Filter," Proc. IEEE (Letters) 55, 688-689 (1967), DDC AD-658784.
10. J. B. Knowles and E. M. Olcayto, "Coefficient Accuracy and Digital Filter Response," IEEE Trans. Circuit Theory CT-15, 31-41 (1968).
11. P. E. Mantey, "Eigenvalue Sensitivity and State-Variable Selection," IEEE Trans. Automat. Control AC-13, No. 3, 263-269 (1968).
12. C. Weinstein, "Quantization Effects in Frequency Sampling Filters," NEREM Record (1968), pp. 222-223, DDC AD-698573.
13. W. R. Bennett, "Spectra of Quantized Signals," Bell System Tech. J. 27, 446-472 (1948).
14. B. Widrow, "Statistical Analysis of Amplitude-Quantized Sampled-Data Systems," AIEE Trans. Applications and Industry, No. 52, 555-568 (January 1961).
15. J. B. Knowles and R. Edwards, "Autocorrelation Function of the Quantization Error Process for a Sinusoid," private communication.
16. O. Sornmoonpin, "Investigation of Quantization Errors," M. S. dissertation, University of Manchester Institute of Science and Technology (May 1966).
17. J. H. Wilkinson, Rounding Errors in Algebraic Processes (Prentice-Hall, Englewood Cliffs, New Jersey, 1963).
18. U. Grenander and M. Rosenblatt, Statistical Analysis of Stationary Time Series (Wiley, New York, 1957).
19. J. W. Cooley, P. A. W. Lewis and P. D. Welch, "The Fast Fourier Transform Algorithm and Its Applications," IBM Research paper RC-1743 (February 1968).
20. J. B. Knowles and R. Edwards, "Effect of a Finite-Word-Length Computer in a Sampled-Data Feedback System," Proc. IEE (London) 112, 1197-1207 (1965).
21. B. Gold and C. M. Rader, "Effect of Quantization Noise in Digital Filters," Proceedings Spring Joint Computer Conference, Vol. 28 (1966), pp. 213-219.
22. B. Gold and L. R. Rabiner, "Analysis of Digital and Analog Formant Synthesizers," IEEE Trans. Audio Electroacoust. AU-16, 81-94 (1968), DDC AD-673594.
23. I. W. Sandberg, "Floating-Point-Roundoff Accumulation in Digital Filter Realizations," Bell System Tech. J. 46, 1775-1791 (1967).
24. T. Kaneko and B. Liu, "Round-off Error of Floating-Point Digital Filters," Proceedings of the Sixth Annual Allerton Conference on Circuit and System Theory, 2-4 October 1968.
25. C. J. Weinstein and A. V. Oppenheim, "A Comparison of Roundoff Noise in Floating Point and Fixed Point Digital Filter Realizations," Proc. IEEE (Letters) 57, 1181-1183 (1969), DDC AD-693897.

26. A. V. Oppenheim, "Block-Floating-Point Realization of Digital Filters," Technical Note 1969-19, Lincoln Laboratory, M. I. T. (20 March 1969), DDC AD-685698.

27. P. D. Welch, "A Fixed-Point Fast Fourier Transform Error Analysis," IEEE Trans. Audio Electroacoust. AU-17, No. 2, 151-157 (1969).

28. W. M. Gentleman and G. Sande, "Fast Fourier Transforms – For Fun and Profit," Proceedings Fall Joint Computer Conference (1966), pp. 563-578.

29. C. J. Weinstein, "Roundoff Noise in Floating Point Fast Fourier Transform Computation," IEEE Trans. Audio Electroacoust. AU-17, No. 3, 209-215 (1969), DDC AD-703354.

30. A. V. Oppenheim and C. Weinstein, "A Bound on the Output of a Circular Convolution with Application to Digital Filtering," IEEE Trans. Audio Electroacoust. AU-17, 120-124 (1969), DDC AD-696240.

31. R. B. Blackman, Linear Data – Smoothing and Prediction in Theory and Practice (Addison-Wesley, Reading, Massachusetts, 1965).

32. B. Gold and K. L. Jordan, Jr., "A Note on Digital Filter Synthesis," Proc. IEEE (Letters) 56, 1717-1718 (1968), DDC AD-688555.

33. K. L. Jordan, Jr. and B. Gold, "A Direct Search Procedure for Designing Finite Duration Impulse Response Filters," IEEE Trans. Audio Electroacoust. AU-17, 33-36 (1969), DDC AD-694141.

DOCUMENT CONTROL DATA – R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Lincoln Laboratory, M.I.T.	2a. REPORT SECURITY CLASSIFICATION Unclassified
	2b. GROUP None

3. REPORT TITLE

Quantization Effects in Digital Filters

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Technical Report

5. AUTHOR(S) *(Last name, first name, initial)*

Weinstein, Clifford J.

6. REPORT DATE 21 November 1969	7a. TOTAL NO. OF PAGES 96	7b. NO. OF REFS 33

8a. CONTRACT OR GRANT NO. AF 19(628)-5167	9a. ORIGINATOR'S REPORT NUMBER(S) Technical Report 468
b. PROJECT NO. 649L	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* ESD-TR-69-348
c.	
d.	

10. AVAILABILITY/LIMITATION NOTICES

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES None	12. SPONSORING MILITARY ACTIVITY Air Force Systems Command, USAF

13. ABSTRACT

Quantization effects in digital filters can be divided into four main categories: quantization of system coefficients, errors due to A-D conversion, errors due to roundoffs in the arithmetic, and a constraint on signal level due to the requirement that overflow must be prevented in the computation.

The effects of quantization on implementations of two basic algorithms of digital filtering – the first- or second-order linear recursive difference equation, and the fast Fourier transform (FFT) – are studied in some detail. For these algorithms, the differing quantization effects of fixed point, floating point, and block floating point arithmetic are examined and compared.

The ideas developed in the study of simple recursive filters and the FFT are applied to analyze the effects of coefficient quantization, roundoff noise, and the overflow constraint in two more complicated types of digital filters – frequency sampling and FFT filters. Realizations of the same filter design, by means of the frequency sampling and FFT methods, are compared on the basis of differing quantization effects.

All the noise analyses in this report are based on simple statistical models for roundoff and A-D conversion errors. Experimental noise measurements testing the predictions of these models are reported, and the empirical results are generally in good agreement with the statistical predictions.

14. KEY WORDS

quantization effects	digital filters	fast Fourier transforms
FFT filters	analog-to-digital conversion	noise-to-signal ratios
recursive filters		frequency sampling filters